Southbridge
+
СЛЁРМ

Тестирование infrastructure as Code

slurm.io

Как вы вносите изменения в Ansible роли?

Процесс изменения роли

1. *Поднимаем виртуалку*
2. *Меняем роль*
3. *Прогоняем на виртуалке*
4. *Чиним что ломалось*
5. *Повторяем пункты 3-4*

Процесс изменения роли

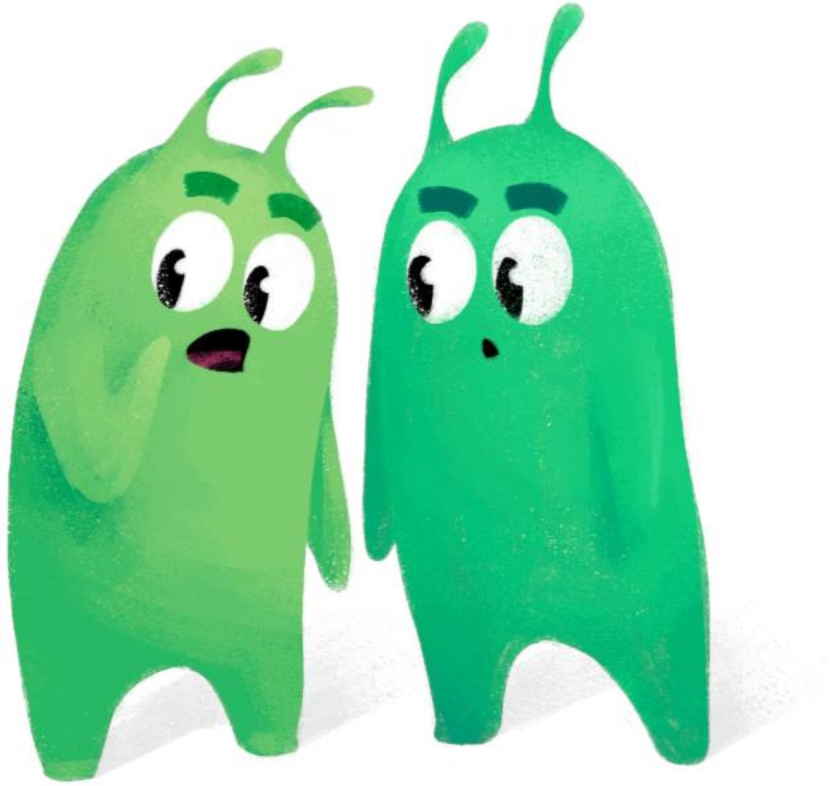
1. *Поднимаем виртуалку*
2. *Меняем роль*
3. *Прогоняем на виртуалке*
4. *Чиним что ломалось*
5. *Повторяем пункты 3-4*
6. *Создаем новую виртуалку, что проверить роль с нуля*
7. *Повторяем пункты 3-6*

Процесс изменения роли

1. *Поднимаем виртуалку*
2. *Меняем роль*
3. *Прогоняем на виртуалке*
4. *Чиним что ломалось*
5. *Повторяем пункты 3-4*
6. *Создаем новую виртуалку, что проверить роль с нуля*
7. *Повторяем пункты 3-6*
8. *Да ну его в баню, погнали, проверим на стейдже/проде*

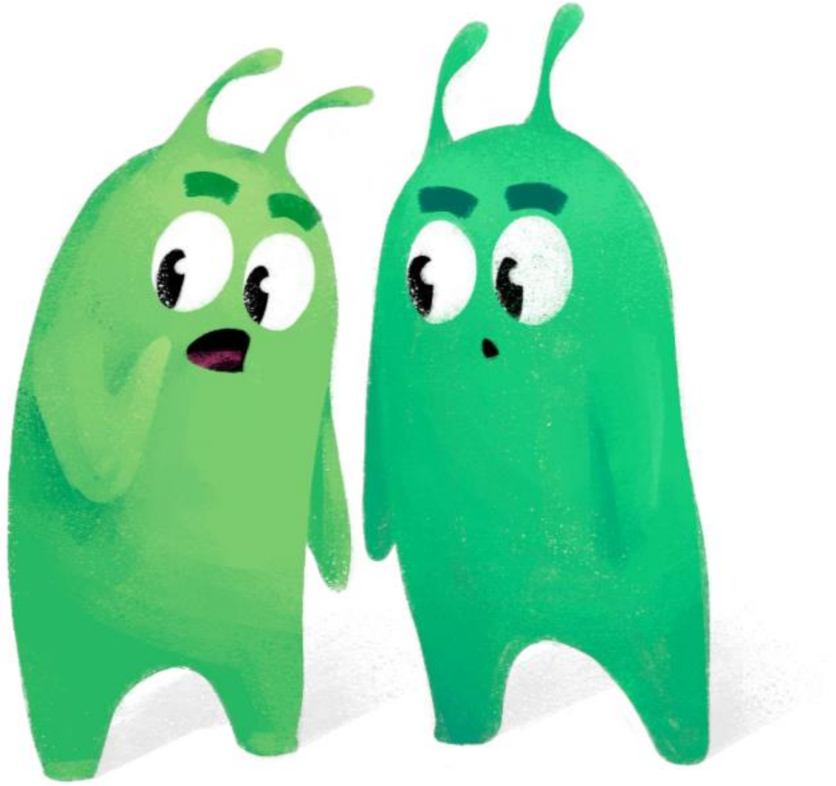
Проблема Ansible

- *Идемпотентность*



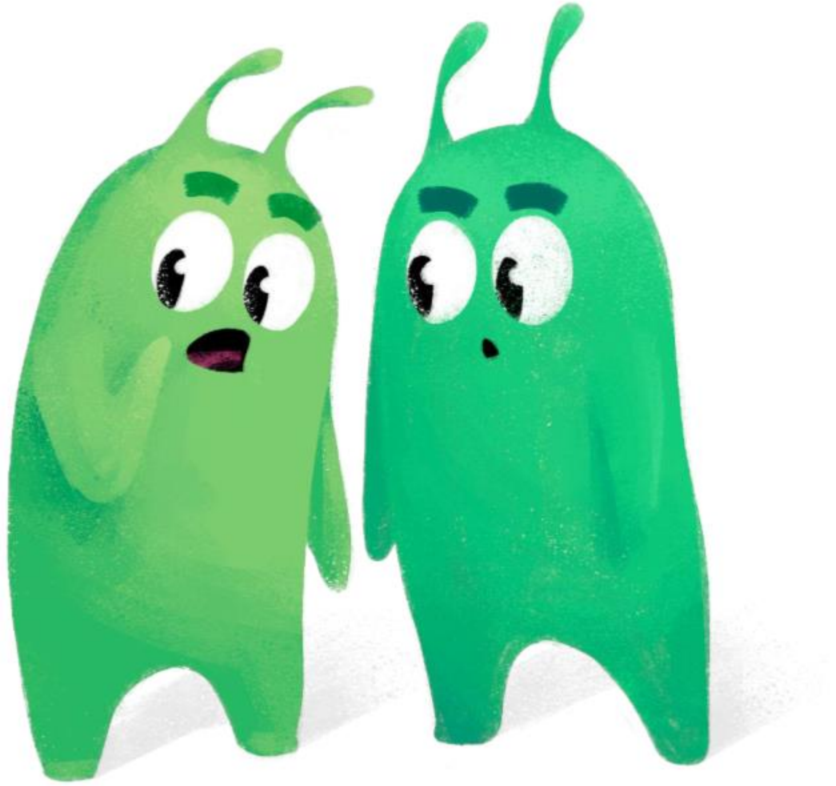
Проблема Ansible

- *Идемпотентность*
- *Декларативность (?)*



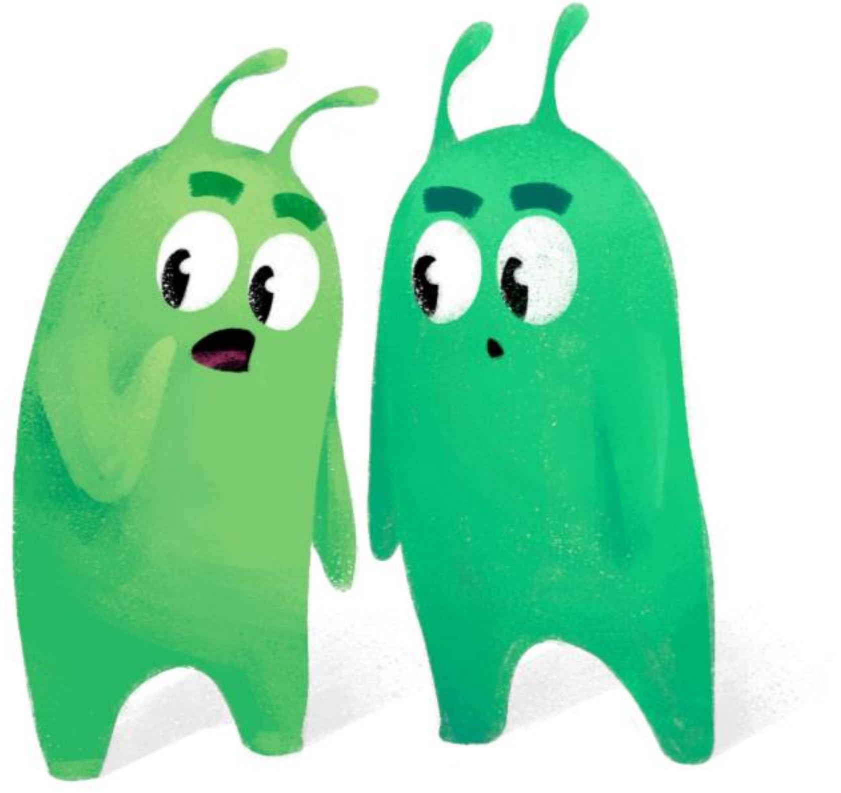
Проблема Ansible

- *Идемпотентность*
- *Декларативность (?)*
- *Несколько синтаксисов*



Проблема Ansible

- *Идемпотентность*
- *Декларативность (?)*
- *Несколько синтаксисов*
- *Плохо воспроизводимое тестирование*

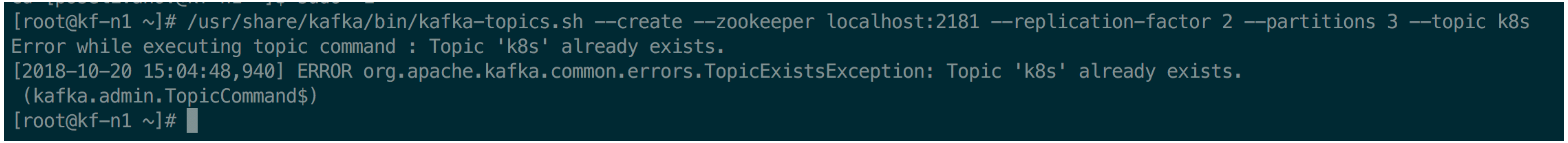


Идемпотентность

```
-   name: Create k8s topic
    shell: /usr/share/kafka/bin/kafka-topics.sh
    --create
    --zookeeper localhost:2181
    --replication-factor 2
    --partitions 3
    --topic k8s
```

Идемпотентность

```
- name: Create k8s topic
  shell: /usr/share/kafka/bin/kafka-topics.sh
  --create
  --zookeeper localhost:2181
  --replication-factor 2
  --partitions 3
  --topic k8s
```



```
[root@kf-n1 ~]# /usr/share/kafka/bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 2 --partitions 3 --topic k8s
Error while executing topic command : Topic 'k8s' already exists.
[2018-10-20 15:04:48,940] ERROR org.apache.kafka.common.errors.TopicExistsException: Topic 'k8s' already exists.
 (kafka.admin.TopicCommand$)
[root@kf-n1 ~]#
```

Идемпотентность

```
-   name: Create k8s topic
    shell: /usr/share/kafka/bin/kafka-topics.sh
    --create
    --zookeeper localhost:2181
    --replication-factor 2
    --partitions 3
    --topic k8s
```

```
/usr/share/kafka/bin/kafka-topics.sh
    --create
    --zookeeper localhost:2181
    --replication-factor 2
    --partitions 3
    --topic k8s
    --if-not-exists
```

Декларативность

- В основе Ansible лежит понятие **Task**
  - **НО!** Возможна декларативность на уровне пользователя playbooka

Виды тестов

Тип тестов	Преимущества	Недостатки
Lint	1. Быстрые, 2. стабильные, 3. не требуется разворачивать реальную инфу, 4. простые.	1. Ограниченное количество ошибок, 2. не тестируется логика.
Unit-test	1. Достаточно быстрые (1-10 мин), 2. достаточно стабильны, 3. тестирование логики (отдельный unit).	1. Нужна реальная инфраструктура, 2. написание тестов - нетривиальная задача.
Integration test	1. Достаточно стабильны, 2. тестирование логики.	1. Нужна реальная инфраструктура, 2. написание тестов - нетривиальная задача, 3. медленный (30-60 минут).
e2e	1. Тестирование архитектуры в целом.	1. Нужна реальная инфраструктура, 2. написание тестов - нетривиальная задача, 3. медленный (60-240+ минут), 4. нестабильные.

Линт решает проблемы

- Быстрая проверка ошибок синтаксиса
- Общий синтаксис включая Best Practice
- Устаревшие конструкции
- Поведение роли

Линтеры для Ansible

- **yamllint**

  https://github.com/adrienverge/yamllint
- **ansible-lint**

  https://github.com/willthames/ansible-lint
- **ansible-playbook --syntax-check**

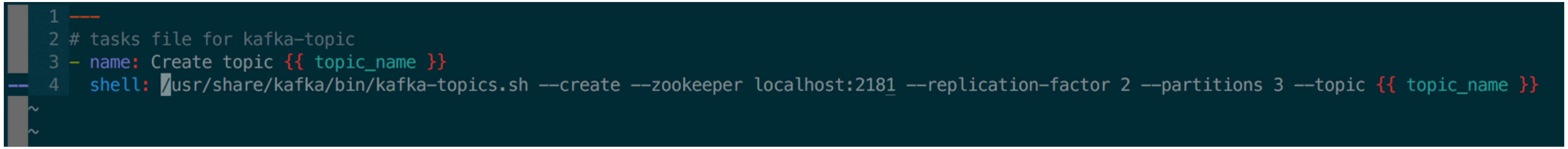
Правила линта

- **Длинна строки ≯ 80**

Правила линта

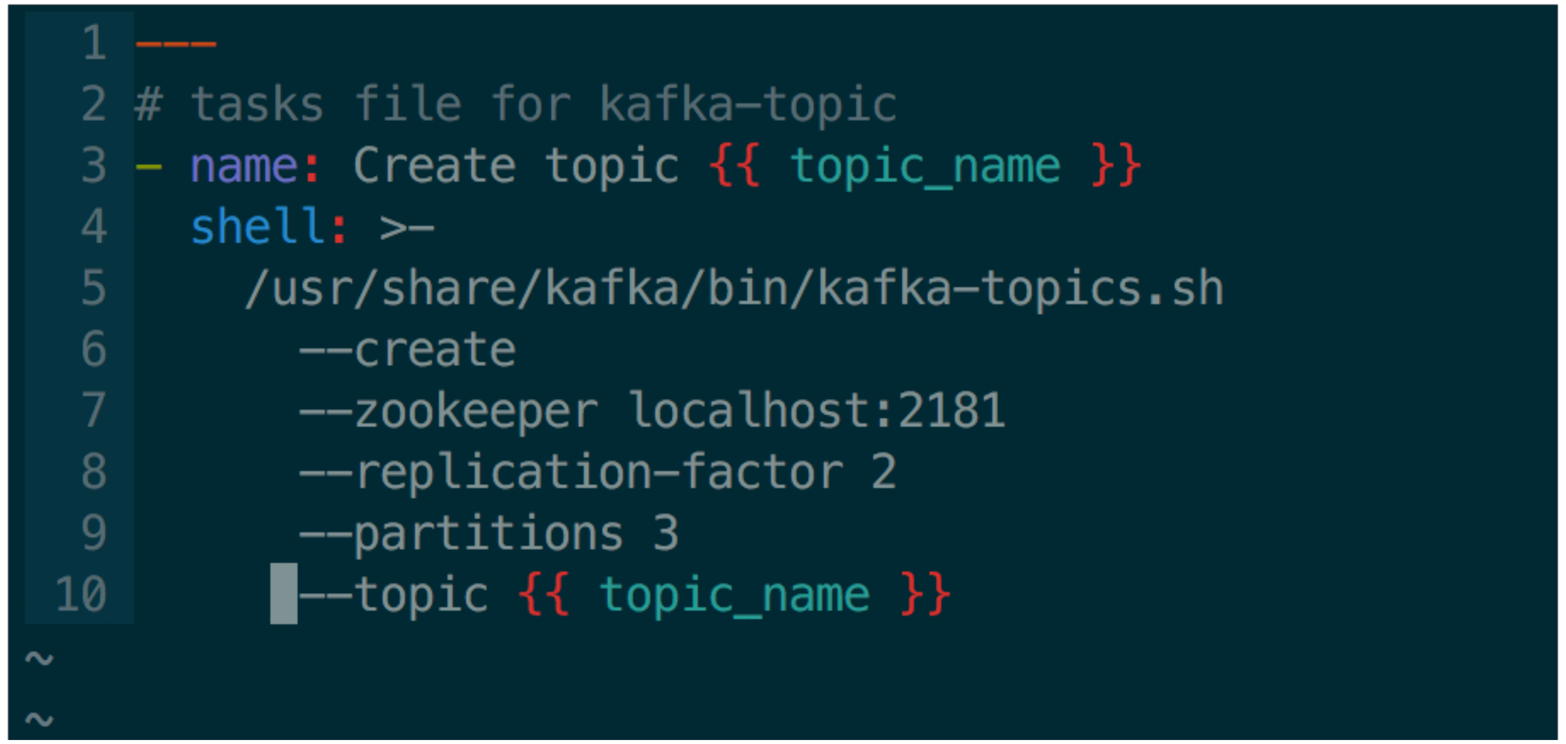
- **Длинна строки ≯ 80**
  - Читаемость
  - Удобство редактирование нескольких документов параллельно
  - Свойство человеческого восприятия

Правила линта



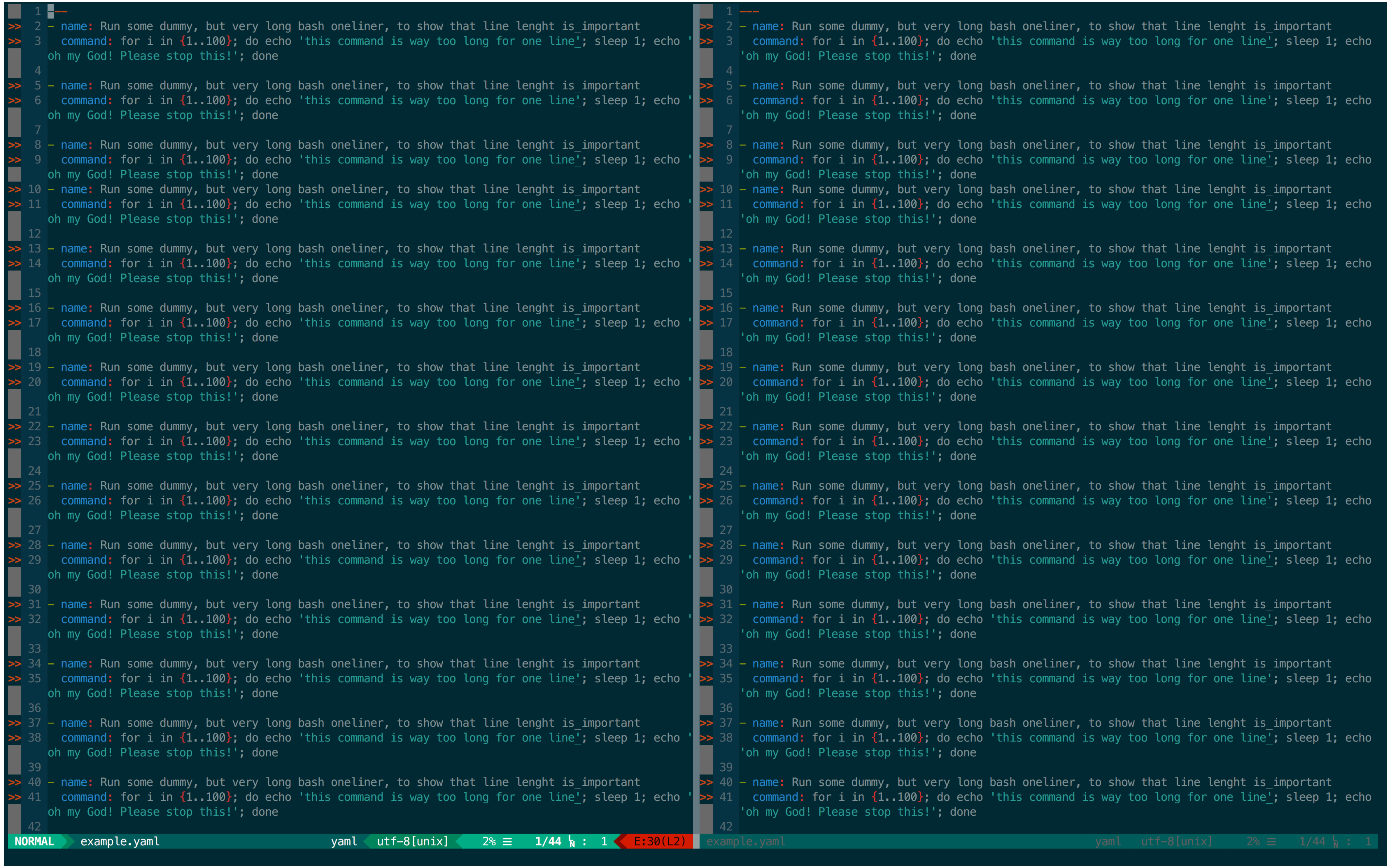
```
---
# tasks file for kafka-topic
- name: Create topic {{ topic_name }}
  shell: /usr/share/kafka/bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 2 --partitions 3 --topic {{ topic_name }}
~
~
```

VS



```
---
# tasks file for kafka-topic
- name: Create topic {{ topic_name }}
  shell: >-
    /usr/share/kafka/bin/kafka-topics.sh
      --create
      --zookeeper localhost:2181
      --replication-factor 2
      --partitions 3
      --topic {{ topic_name }}
~
~
```

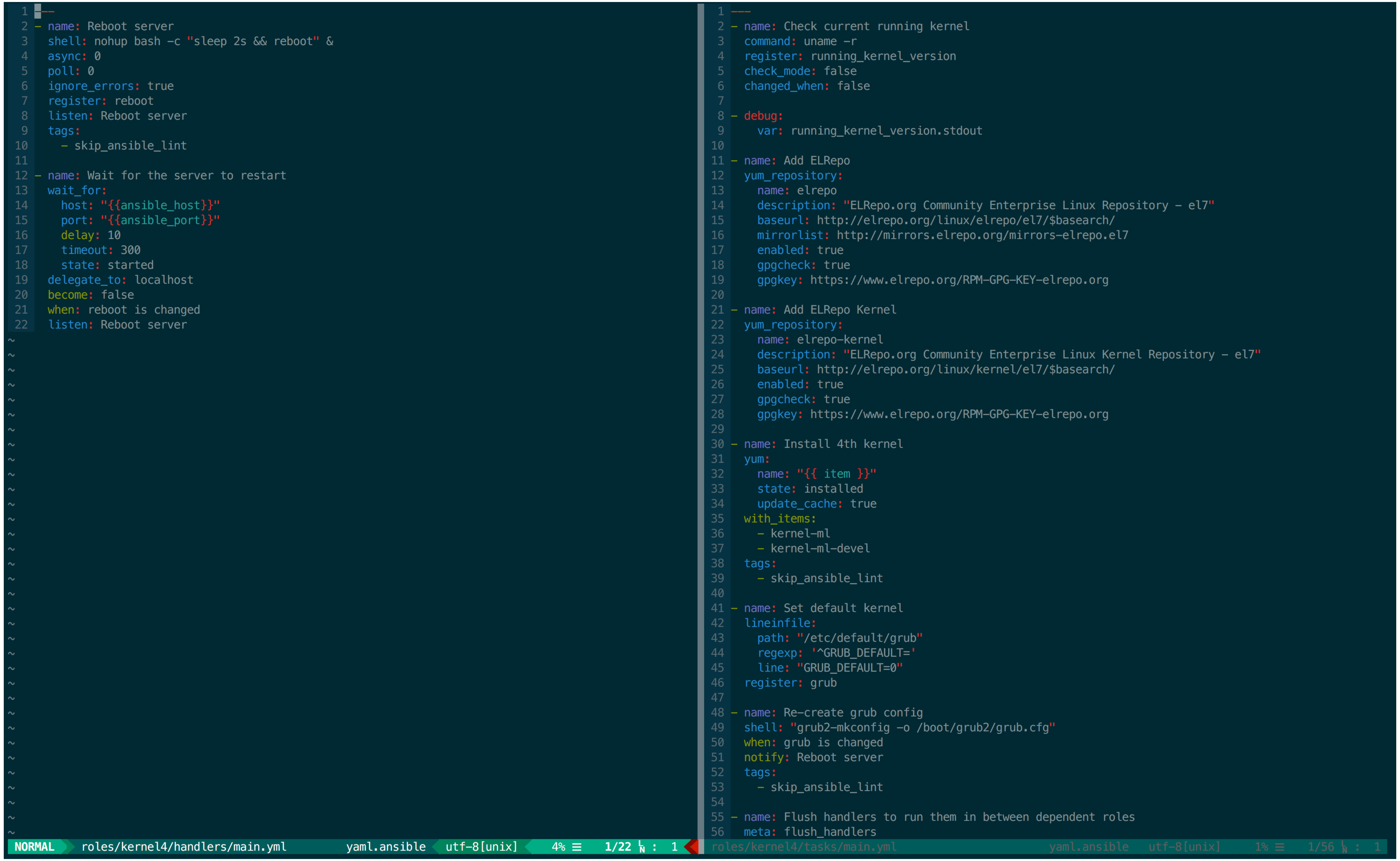
Правила линта



```
   ---
>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo '
   oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo '
   oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo '
   oh my God! Please stop this!'; done
>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo '
   oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo '
   oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo '
   oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo '
   oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo '
   oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo '
   oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo '
   oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo '
   oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo '
   oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo '
   oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo '
   oh my God! Please stop this!'; done

 NORMAL  example.yaml                          yaml  utf-8[unix]  2% ≡  1/44 ㏑ :  1  E:30(L2)
```

```
   ---
>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo
   'oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo
   'oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo
   'oh my God! Please stop this!'; done
>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo
   'oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo
   'oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo
   'oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo
   'oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo
   'oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo
   'oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo
   'oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo
   'oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo
   'oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo
   'oh my God! Please stop this!'; done

>> - name: Run some dummy, but very long bash oneliner, to show that line lenght is important
>>   command: for i in {1..100}; do echo 'this command is way too long for one line'; sleep 1; echo
   'oh my God! Please stop this!'; done

 example.yaml                                        yaml   utf-8[unix]   2% ≡   1/44 ㏑ :  1
```

Правила линта



```
---
- name: Reboot server
  shell: nohup bash -c "sleep 2s && reboot" &
  async: 0
  poll: 0
  ignore_errors: true
  register: reboot
  listen: Reboot server
  tags:
    - skip_ansible_lint

- name: Wait for the server to restart
  wait_for:
    host: "{{ansible_host}}"
    port: "{{ansible_port}}"
    delay: 10
    timeout: 300
    state: started
  delegate_to: localhost
  become: false
  when: reboot is changed
  listen: Reboot server
```

```
NORMAL  roles/kernel4/handlers/main.yml        yaml.ansible  utf-8[unix]   4% ≡   1/22 ㏑ :  1
```

```
---
- name: Check current running kernel
  command: uname -r
  register: running_kernel_version
  check_mode: false
  changed_when: false

- debug:
    var: running_kernel_version.stdout

- name: Add ELRepo
  yum_repository:
    name: elrepo
    description: "ELRepo.org Community Enterprise Linux Repository - el7"
    baseurl: http://elrepo.org/linux/elrepo/el7/$basearch/
    mirrorlist: http://mirrors.elrepo.org/mirrors-elrepo.el7
    enabled: true
    gpgcheck: true
    gpgkey: https://www.elrepo.org/RPM-GPG-KEY-elrepo.org

- name: Add ELRepo Kernel
  yum_repository:
    name: elrepo-kernel
    description: "ELRepo.org Community Enterprise Linux Kernel Repository - el7"
    baseurl: http://elrepo.org/linux/kernel/el7/$basearch/
    enabled: true
    gpgcheck: true
    gpgkey: https://www.elrepo.org/RPM-GPG-KEY-elrepo.org

- name: Install 4th kernel
  yum:
    name: "{{ item }}"
    state: installed
    update_cache: true
  with_items:
    - kernel-ml
    - kernel-ml-devel
  tags:
    - skip_ansible_lint

- name: Set default kernel
  lineinfile:
    path: "/etc/default/grub"
    regexp: '^GRUB_DEFAULT='
    line: "GRUB_DEFAULT=0"
  register: grub

- name: Re-create grub config
  shell: "grub2-mkconfig -o /boot/grub2/grub.cfg"
  when: grub is changed
  notify: Reboot server
  tags:
    - skip_ansible_lint

- name: Flush handlers to run them in between dependent roles
  meta: flush_handlers
```

```
roles/kernel4/tasks/main.yml        yaml.ansible   utf-8[unix]   1% ≡   1/56 ㏑ :  1
```

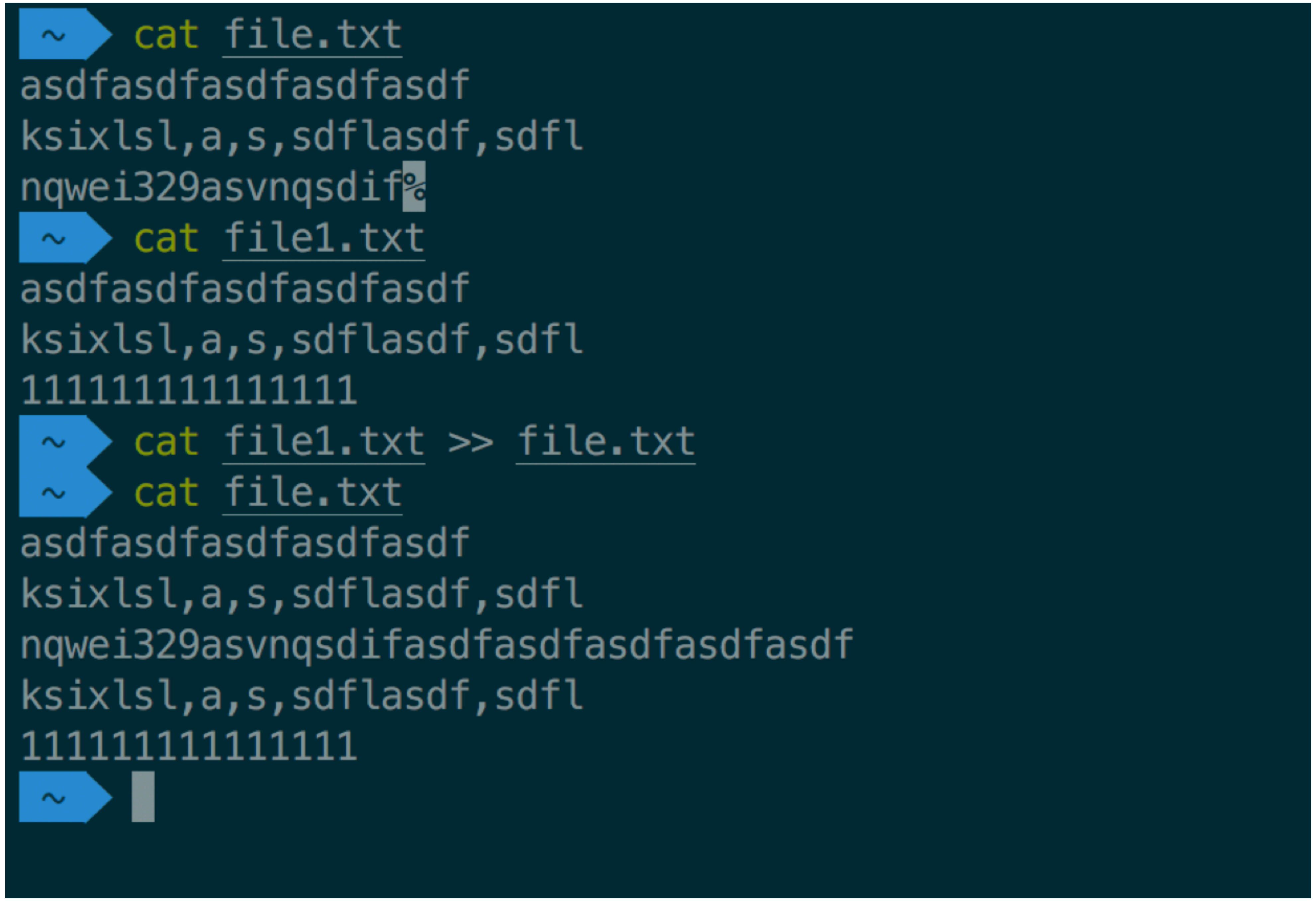
Правила линта

- **Файл должет заканчиваться переводом строки**

Правила линта

- **Файл должет заканчиваться переводом строки**
  - Корректность работы GNU утилит

Правила линта



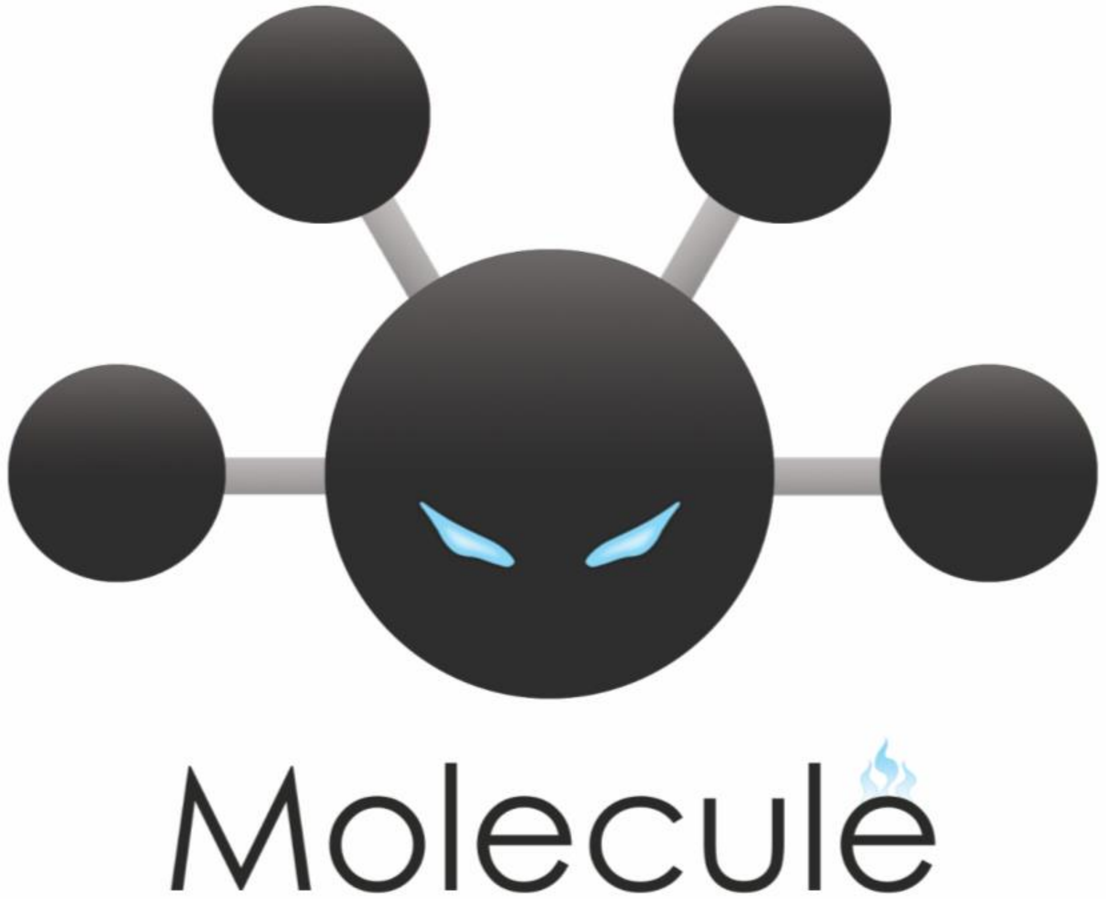
```
~ cat file.txt
asdfasdfasdfasdfasdf
ksixlsl,a,s,sdflasdf,sdfl
nqwei329asvnqsdif%
~ cat file1.txt
asdfasdfasdfasdfasdf
ksixlsl,a,s,sdflasdf,sdfl
111111111111111
~ cat file1.txt >> file.txt
~ cat file.txt
asdfasdfasdfasdfasdf
ksixlsl,a,s,sdflasdf,sdfl
nqwei329asvnqsdifasdfasdfasdfasdfasdf
ksixlsl,a,s,sdflasdf,sdfl
111111111111111
~ 
```

Правила линта

- **Бинарные переменные должны принимать занчения true/false**
  - Поиск по документу
  - Единообразие

Тестирование

- Сложно тестить локально
- Нужна виртуальная машина
- Поднимать систему с нуля долго (от 5 минут) + ручные монотонные операции
- Протестировать роль лишний раз на свежей системе влом
- Изменения непредсказуемы


Molecule

Molecule

- Фрэймворк для тестирования Ansible ролей
- Реализует весь необходимый функционал
  - Lint
  - Setup
  - Execute (run + rerun)
  - Test

Почему не пишут тесты:

- *Там же декларативные языки.*
- *Я один разрабатываю, я ничего не сломаю.*
- *Мы только начали разработку, напишем потом*
- *У нас нет времени писать тесты*
- *Мои роли нельзя протестировать*

Ansible believes you should not need another framework to validate basic things of your infrastructure is true.

This is the case because Ansible is an order-based system that will fail immediately on unhandled errors for a host, and prevent further configuration of that host.

This forces errors to the top and shows them in a summary at the end of the Ansible run.

Преимущества использования тестов

- Экономит время и ресурсы.
- Обеспечивает безопасность кода при командно й работе.
- Улучшает качество кода.
- Делает рефакторинг и переход на новые версии ПО простым и безопасным.

СЛЁРМ

slurm.io

Southbridge

southbridge.io