



Фронт Халяль

Борисенко Глеб

ФТиАД2021

(сорян за шакалов)

Что это

- Фронт – то, с помощью чего с вашей приложухой взаимодействует пользователь. Обычно фронт отправляет запросы бэк и получает от него ответы, с помощью которых и отображает нужное пользователю.
- Фронт бывает разный – web, android app, военный, win app, etc.
- Не ссыте, нас конкретно интересует фронт web-приложения.

Web-фронт

- По сути, это просто страничка в интернете.
- А что такое страничка?
- Это (чаще всего, но не всегда):
 - HTML
 - CSS
 - JS
- JS нам не уперся, штука «своеобразная», а остальное посмотрим.

HTML

- HTML – гипертекстовый язык разметки. Основным элементом являются теги.
- Определяет общую содержание и структуру веб-контента, с него все начинается
- Расширяется с помощью CSS (для красивостей) и JS (для анимационных и интерактивных красивостей)

Пример html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Моя тестовая страница</title>
  </head>
  <body>
    
  </body>
</html>
```

Теги

- Как вы увидели, он состоит из тегов (парных и одинарных)
- У каждого тега можно писать дополнительные параметры (а точнее, атрибуты)
- Например, размеры и источник картинки:
 - ``
- Одни из самых важных атрибутов – class и id
- id присваивается только одному html-элементу, а класс может присваиваться нескольким
- Используется для связки со стилем

CSS



(типа шутка)

CSS

- Расшифровывается как каскадные таблицы стилей. По сути просто файлы, в которых расположено описание стилей элементов и классов.
- Для каждого нужного элемента идет описание цветов, шрифтов, паддингов, и т.п.

Пример CSS

```
p {  
    font-family: arial, helvetica, sans-serif;  
}  
h2 {  
    font-size: 20pt;  
    color: red;  
    background: white;  
}  
.note {  
    color: red;  
    background-color: yellow;  
    font-weight: bold;  
}
```

Связь с HTML

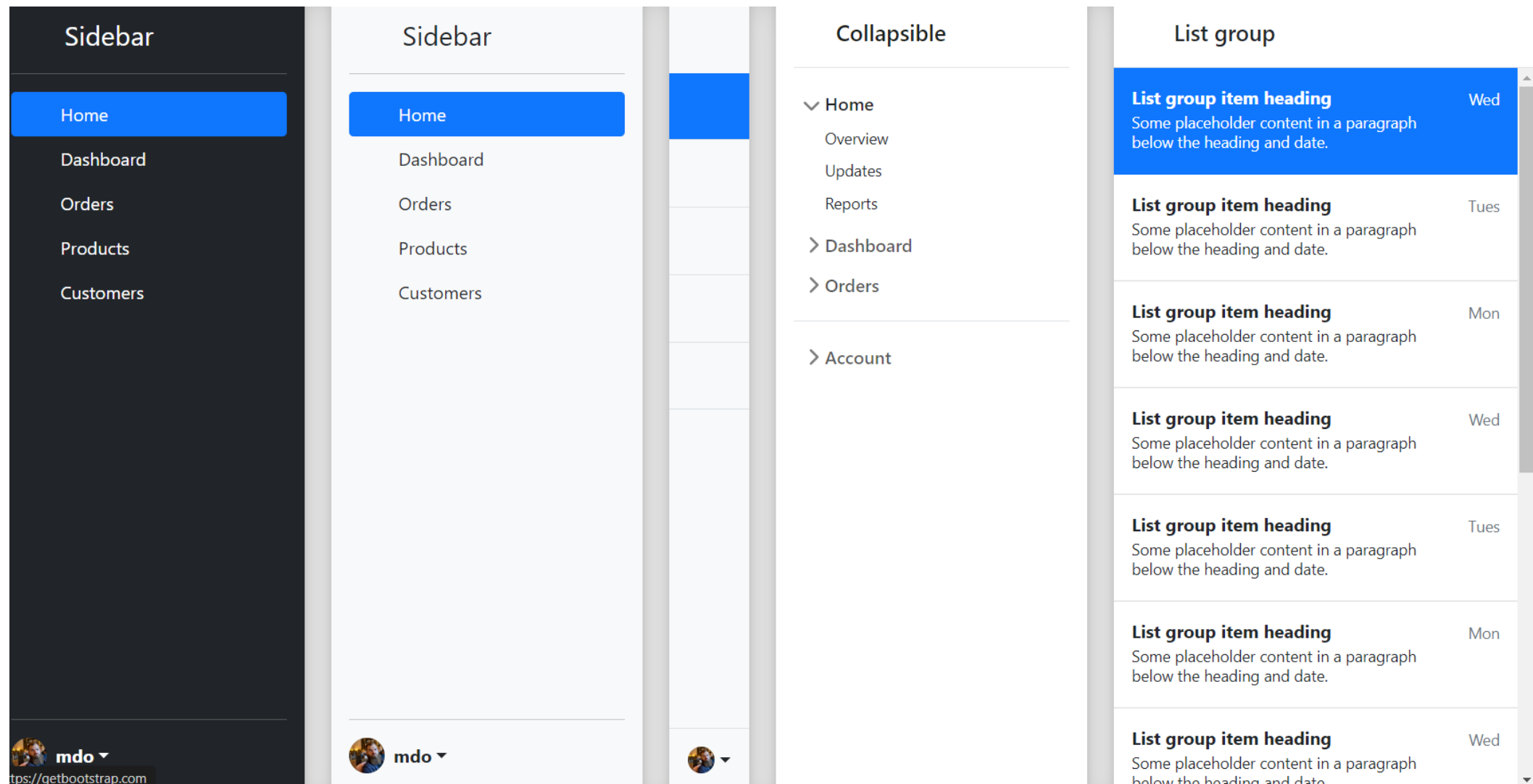
Есть два варианта:

- Через теги `<style>` в голове документа прямым текстом
- Через тег `<link>` в голове, указав расположение файла
- JS если что привязывается похожим образом

Bootstrap

- Фреймворк, позволяющий быстро создавать адаптивный сайт. Включает набор инструментов для создания сайтов и веб-приложений. Содержит HTML и CSS-шаблоны оформления для типографики, веб-форм, кнопок и прочих компонентов веб-интерфейса
- По сути, шаблон и библиотека сниппетов
- Подключить стили и js можно через все так же через `<link>`, в их доке есть куча примеров и все хорошо описано

Например, сайдбары на bootstrap



Использование

- Подключив bootstrap, вам разве что нужно проставить нужные классы у элементов. Усё.

Шаблоны Jinja2

- Шаблон — файл с HTML-кодом и элементами разметки, которые позволяют выводить динамический контент позволяют выводить динамический контент.
- Если говорить о Flask, то у него есть функция `render_template()`, которая рендерит такой шаблон.

Пример шаблона

- Подстановка переменной
- Шаблон:

```
<html>
  <body>
    <h1>Prediction: {{ pred }}</h1>
  </body>
</html>
```

- Рендер шаблона (похоже на format строки питонячи)

```
@app.route('/')
def index():
    return render_template('index.html', pred=model.prediction)
```

Еще пример, с циклом и условием

```
<h1>Members</h1>
<ul>
{% for user in users %}
  <li>{{ user.username|e }}</li>
{% endfor %}
</ul>
```

```
{% if kenny.sick %}
  Kenny is sick.
{% elif kenny.dead %}
  You killed Kenny!  You bastard!!!
{% else %}
  Kenny looks okay --- so far
{% endif %}
```


Возможности шаблонов

- Там вообще куча возможностей, в их доке все есть
- Весьма удобная и прикольная штука
- По сути, мы пишем внутри `{{}}` команды питонячи (с ограничениями вроде), а когда рендерим, они выполняются

Django

- Web-framework для написания бэка на питоне
- Весьма высокоуровневый, удобный
- Для API его будет использовать как пушкой по воробьям
- Но для пет-проджекта или в работе резко понадобится нормальное Web приложение быстро забабахать (такое бывает) – самое то
- У простых людей, не бэков, используется обычно такая связка:
 - Django
 - Bootstrap (один из шаблонов)
 - Jinja2 templating
 - Database (PostgreSQL for example)

В чем суть

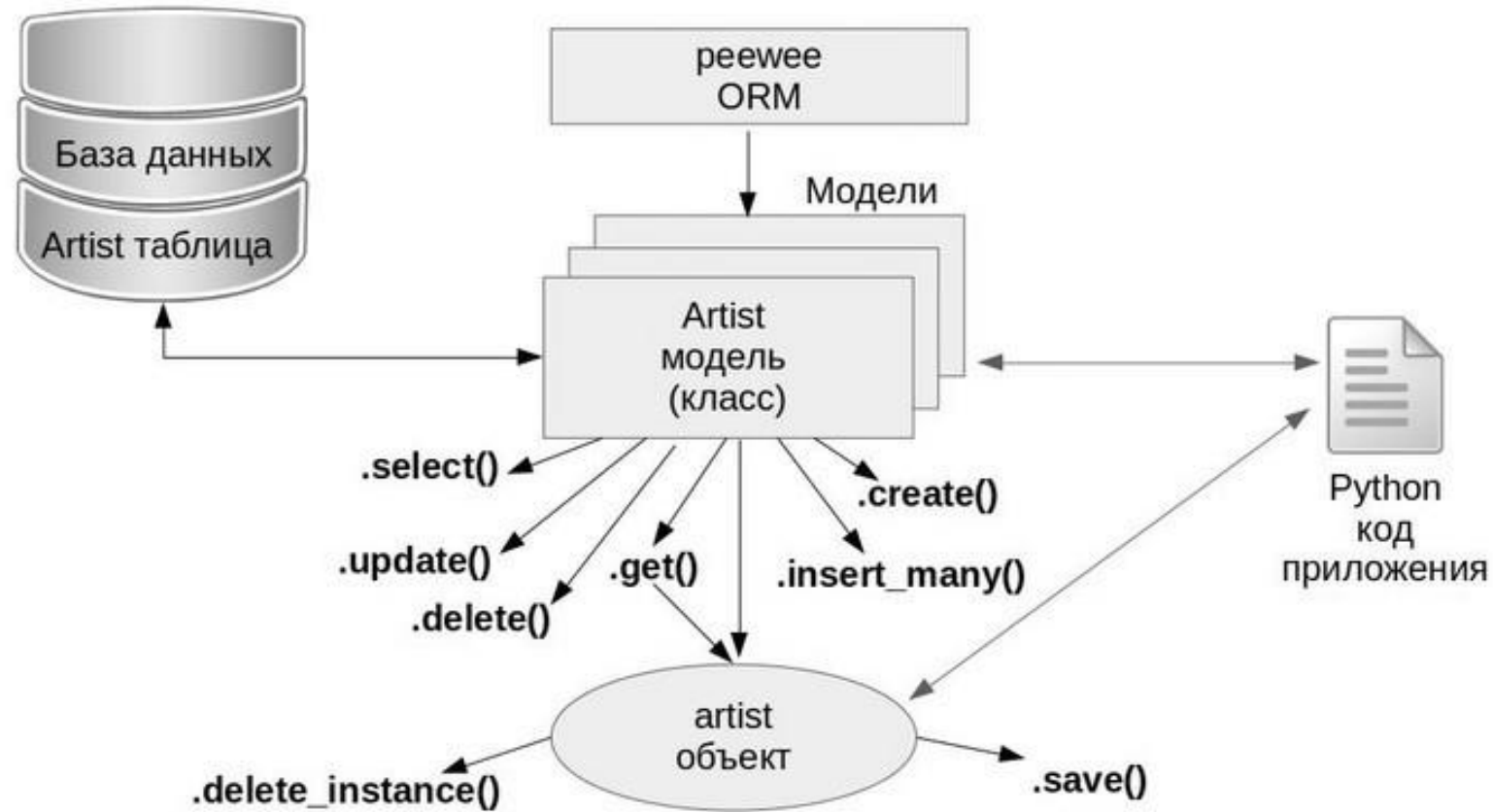
- Вы логику отображения контента пишете в виде view-шек
- В специальном файле `urls.py` связываете свои view-шки с нужными урлами (путями)
- Встроенная работа с БД через ORM и миграции
- Есть кайфовая админ панель из коробки

ORM

Это способ обращения к БД в коде. В чем суть:

- Каждая сущность/таблица/модель – это класс в питоне
- Атрибут сущности/колонка в таблице – это атрибут (параметр) класса в питоне
- Через методы этого класса мы общаемся с конкретной сущностью/таблицей/моделью
- В питоне чаще всего для этого используется SQLAlchemy

Схематично это выглядит так (при использовании либы реееее)



Как это выглядит в Django

```
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, on_delete=models.CASCADE)
    choice_text = models.CharField(max_length=200)
    votes = models.IntegerField(default=0)
```

```
q = Question(question_text="What's new?", pub_date=timezone.now())
q.save()
q = Question.objects.get(pk=1)
```

А как это выглядит при прямом
использовании DB-API (SQLite3)

```
import sqlite3  
con = sqlite3.connect("tutorial.db")  
cur = con.cursor()  
res = cur.execute("SELECT name FROM sqlite_master")  
res.fetchone()
```

Чувствуете разницу, да?

- С ORM работать удобнее и гибче.
- Зачем это вам?
- Вы высоко вероятно будете работать с БД из кода

Streamlit

- Что это? Opensource Python фреймворк для быстрой разработки дашборда в проектах с машинным обучением, не требующий знания frontend (HTML, CSS и JavaScript)
- Зачем? Демонстрировать что-то кому-то, игратья через удобный интерактив с данными, строить легковесные простые дашборды личные

Особенности

- Куча виджетов
- Куча способов визуализации
- Красивый, минималистичный и удобный дизайн
- Легко использовать (серьезно, после семинара можете добавить в резюме, если захотите)

Как это выглядит

×

📄 Hello

📄 Animation Demo

📄 Plotting Demo

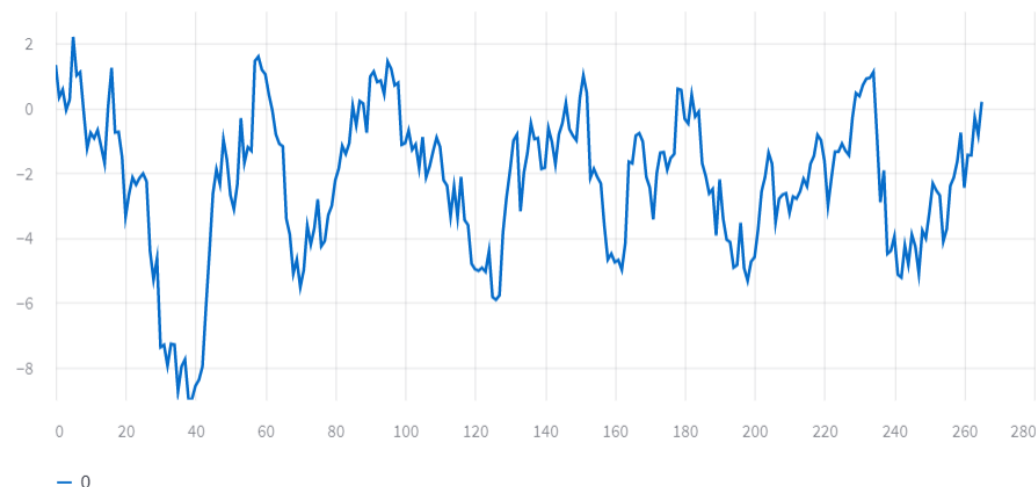
📄 Mapping Demo

Plotting Demo

54% Complete

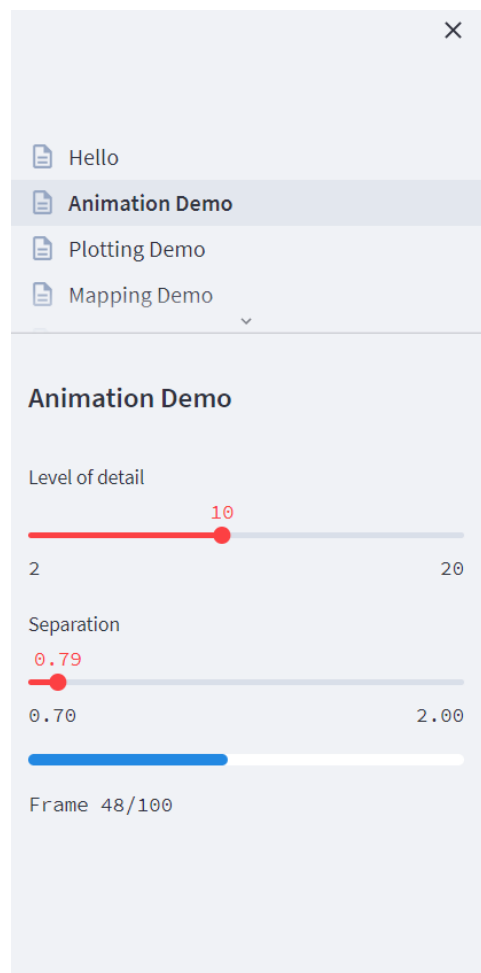
Plotting Demo

This demo illustrates a combination of plotting and animation with Streamlit. We're generating a bunch of random numbers in a loop for around 5 seconds. Enjoy!



 RUNNING... Stop 

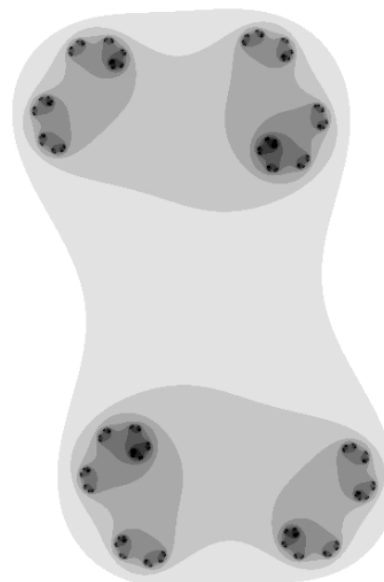
Как это выглядит



🏃 RUNNING... Stop ☰

Animation Demo

This app shows how you can use Streamlit to build cool animations. It displays an animated fractal based on the the Julia Set. Use the slider to tune different parameters.



Как это выглядит в общем



Как это выглядит в коде

- Есть магия (серьезно, они это так и называют – magic)
Вы просто одной строчкой пишете переменную, типа 'df', будто в ячейка юпитера хотите его посмотреть, и он отобразится на страницу
Но это не очень гибко
- Есть методы для отображения и создания чего бы то ни было (типа `streamlit.checkbox` или `streamlit.line_chart`). В каком порядке это выводится в скрипте, в том и будет на страничке
- Отдельный файл в папке `pages` – отдельная страничка в списке на сайдбаре
- Есть возможности конфигурирования `layout`
- Ну и ваш дашборд можно бесплатно задеплоить в их облаке

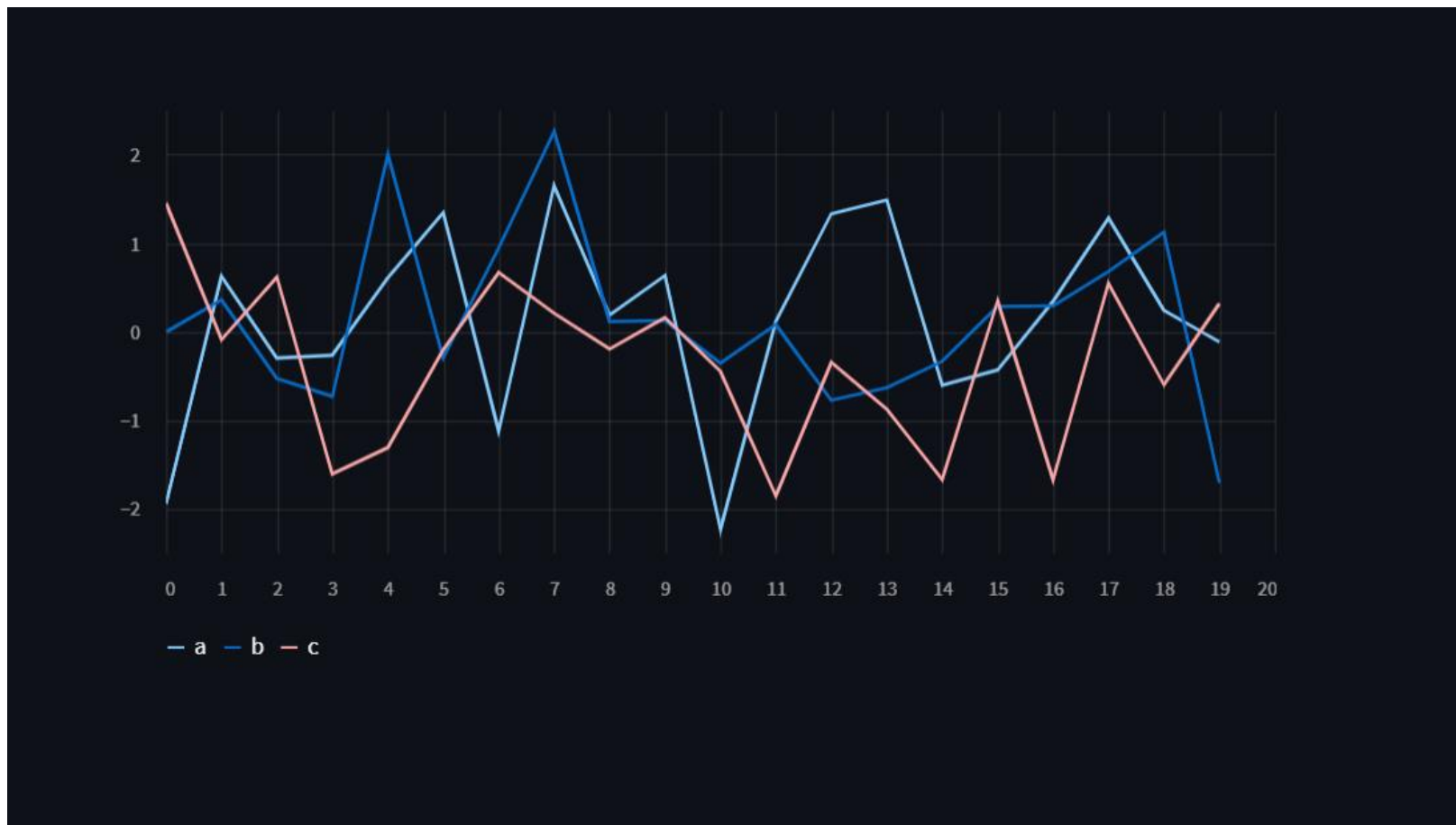
Пример

```
import streamlit as st
import numpy as np
import pandas as pd

chart_data = pd.DataFrame(
    np.random.randn(20, 3),
    columns=['a', 'b', 'c'])

st.line_chart(chart_data)
```

Пример (там можно выбирать тему)



На семинаре

- Расскажу то, что забыл в прошлый раз: про 80 порт, как сделать ваш домашний сервис публичным
- Ручками посмотрим и потрогаем streamlit
- Покажу свою старую штуку на Django
- Любые вопросыки ваши, связанные прямо или косвенно с прошлым материалом
- Посмотрим, что вам захочется

Смотрите в следующей серии

- Вспомним Git
- Рассмотрим классический gitflow
- Посмотрим версионирование артефактов (Nexus, возможно Minio)