Лекция 1
Введение в Web и Django

Разработка интернет приложений

Канев Антон Игоревич

Канев Антон Игоревич

Окончил МГТУ им. Н.Э. Баумана в 2016 г.

Аспирантуру МГТУ им. Н.Э. Баумана в 2020 г.

Окончил магистратуру университет Glyndwr (Рексем, Великобритания)

Курс по глубокому обучению

Курс по web разработке

NVIDIA DLI Certificate

РосЕвроБанк
Совкомбанк

Руководитель проектов

Московский государственный технический университет имени Н.Э. Баумана

Оценивание и сроки

- Экзамен
- 2 рубежных контроля
- Практические задания – закрепление и использование знаний разных дисциплин

- Оценивание – баллы за задания
- Сроки!!!
- Занятия, консультации и вопросы в чате ИУ5 Stack Overflow

Одна тема на весь курс

- Набор требований по каждому заданию – демонстрируется и защищается отдельно
- 8 лабораторных + GitHub + UML
- ТЗ
- ДЗ
- Отчет-РПЗ

- Знание браузера, умение использовать необходимые инструменты
- Ответы на вопросы по базовым понятиям и технологиям

Стек технологий

- React
- Django или Go. Другой бэкенд только по согласованию с преподавателем
- PostgreSQL

- GitHub - репозитории для фронтенда, бэкенда, нативного приложения. Вы работаете на свое портфолио
- VS Code – основная среда разработки
- Виртуальная машина с Ubuntu

Активность вне курса

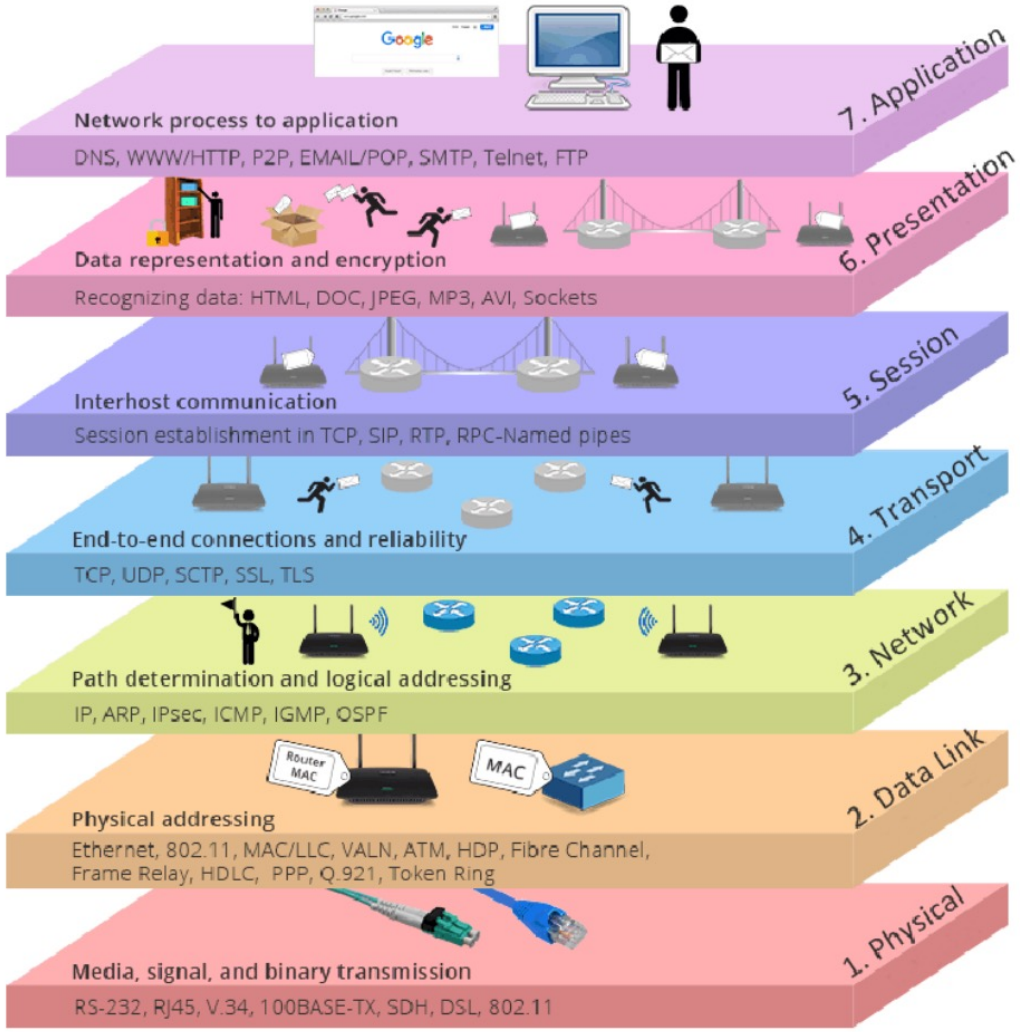
- Хакатон
- Курсы
- Стажировки
- Работа в МГТУ

Стандарты интернета

- В отличие от корпоративных систем, интернет изначально строится на открытых стандартах. Эти стандарты открыто опубликованы, любое заинтересованное лицо может принять участие в их разработке.
- Разработкой стандартов занимается IETF
  - Официальный сайт https://www.ietf.org
  - Список RFC опубликован здесь https://www.rfc-editor.org/rfc-index.html
- Стандарты для URL, HTTP, FTP.

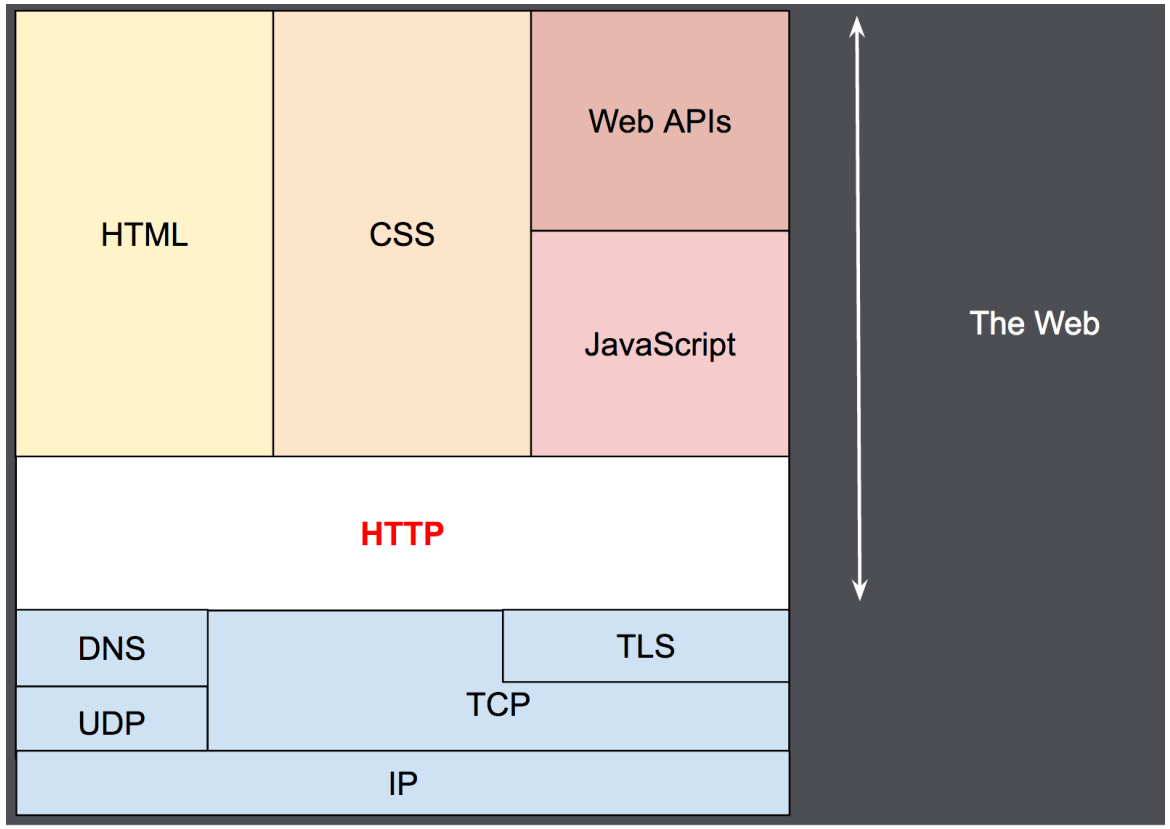
Компьютерные сети. Модель OSI

- 7-ми уровневая модель OSI
- Приложения работают на самом высоком 7-ом уровне
- Физическая среда передачи на первом уровне



Web

- Стандарты Web публикуются на сайте веб-консорциума
- https://www.w3.org



Компоненты Web

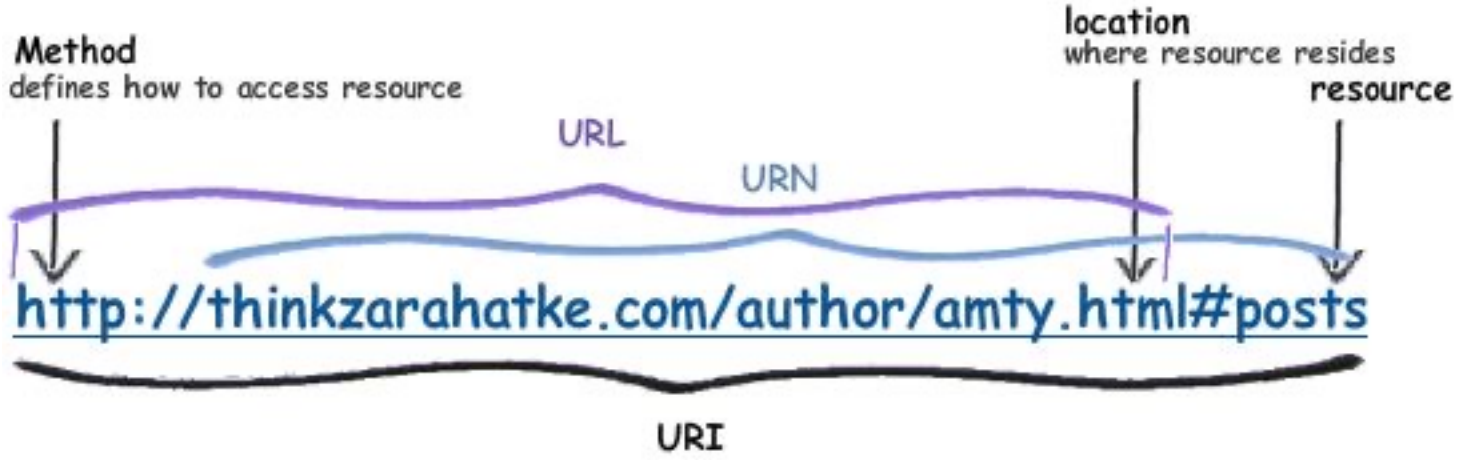
- Тим Бернерс-Ли создал три основных компонента WWW:
- язык гипертекстовой разметки документов HTML (HyperText Markup Language);
- универсальный способ адресации ресурсов URI (Universal [Uniform] Resource Identifier);
- протокол обмена гипертекстовой информацией HTTP (HyperText Transfer Protocol – протокол передачи гипертекста).
- Позже к этим трем компонентам добавился четвертый CGI: исполняемая часть, с помощью которой можно создавать динамические HTML-документы.

HTML

- HTML-HyperText Markup Language.
- В HTML версии 1.0 были реализованы все элементы разметки, связанные с выделением параграфов, шрифтов, стилей и т.п., т.к. уже первая реализация подразумевала графический интерфейс. Важным компонентом языка стало описание гипертекстовых ссылок, графики и обеспечение возможности поиска по ключевым словам.
- В качестве базы для разработки языка гипертекстовой разметки HTML был выбран SGML (Standard Generalised Markup Language – стандартный общий язык разметки). Тим Бернерс-Ли описал HTML в терминах SGML как описывают языки программирования в терминах формы Бекуса-Наура.

URI

- Вторым важным компонентом WWW стал универсальный способ адресации ресурсов URI (Universal Resource Identifier).
- Кроме термина URI можно также встретить термины:
  - URL (Universal Resource Locator),
  - URN (Universal Resource Name).
- Наиболее общим термином является URI, который может быть или URL или URN. В соответствии со спецификацией URL определяет ресурс по механизму доступа к ресурсу, а URN по уникальному имени (это не имя файла).
- В результате терминологической путаницы термины URI и URL часто стали использоваться как синонимы. Термин URN используется достаточно редко. Некоторое применение он нашел в технологии XML.



URI – схема HTTP

- http:// хост : порт / путь и имя файла ? параметры # якорь гиперссылки

- Пример:

http:// 127.0.0.1 :8080/index.html

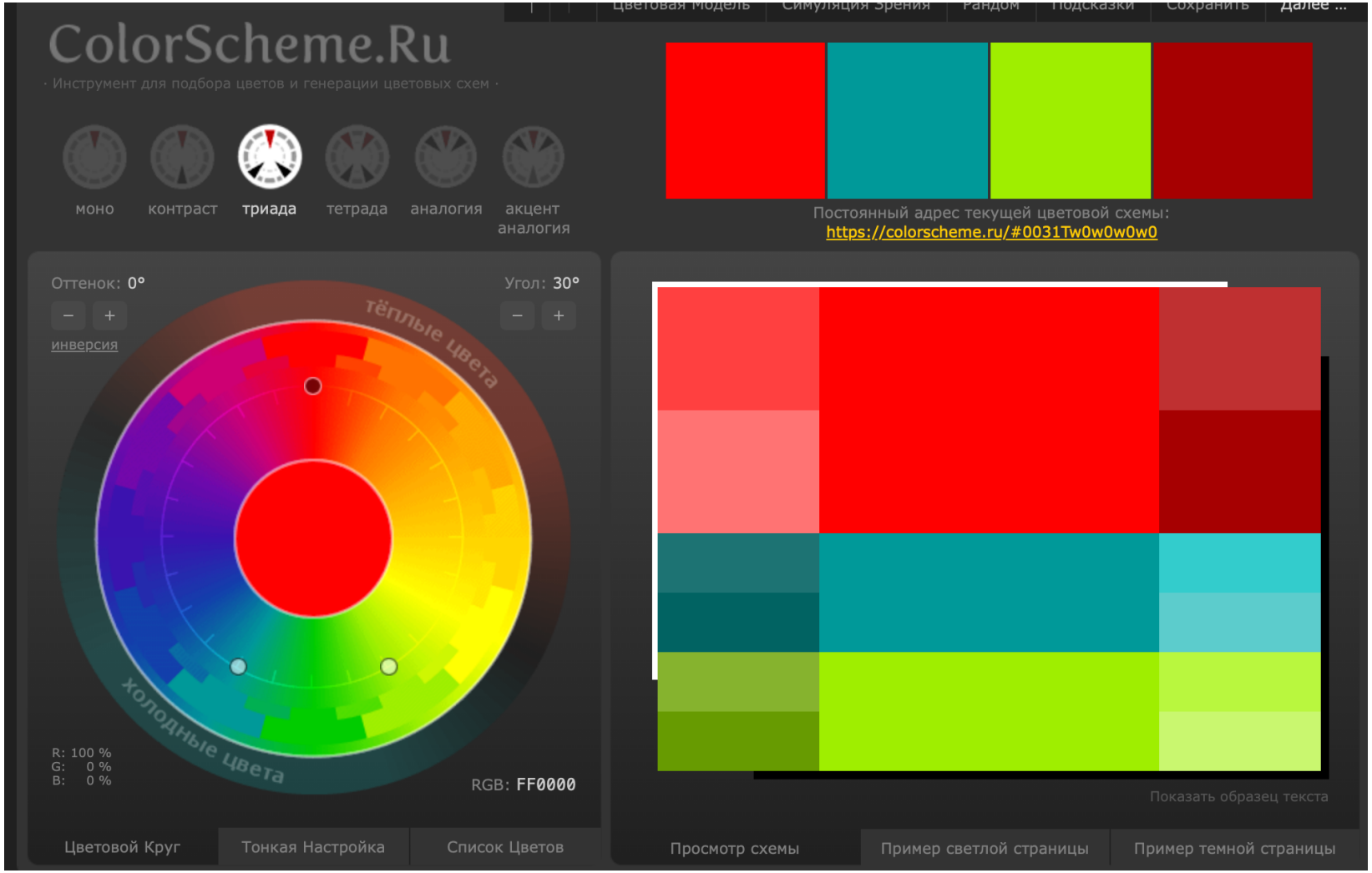
http://localhost:8080/file.html

http://iu5.bmstu.ru:8080/cat1/cat2/script.asp?param1=1&param2=2#anchor1

- Порт по умолчанию – 80.

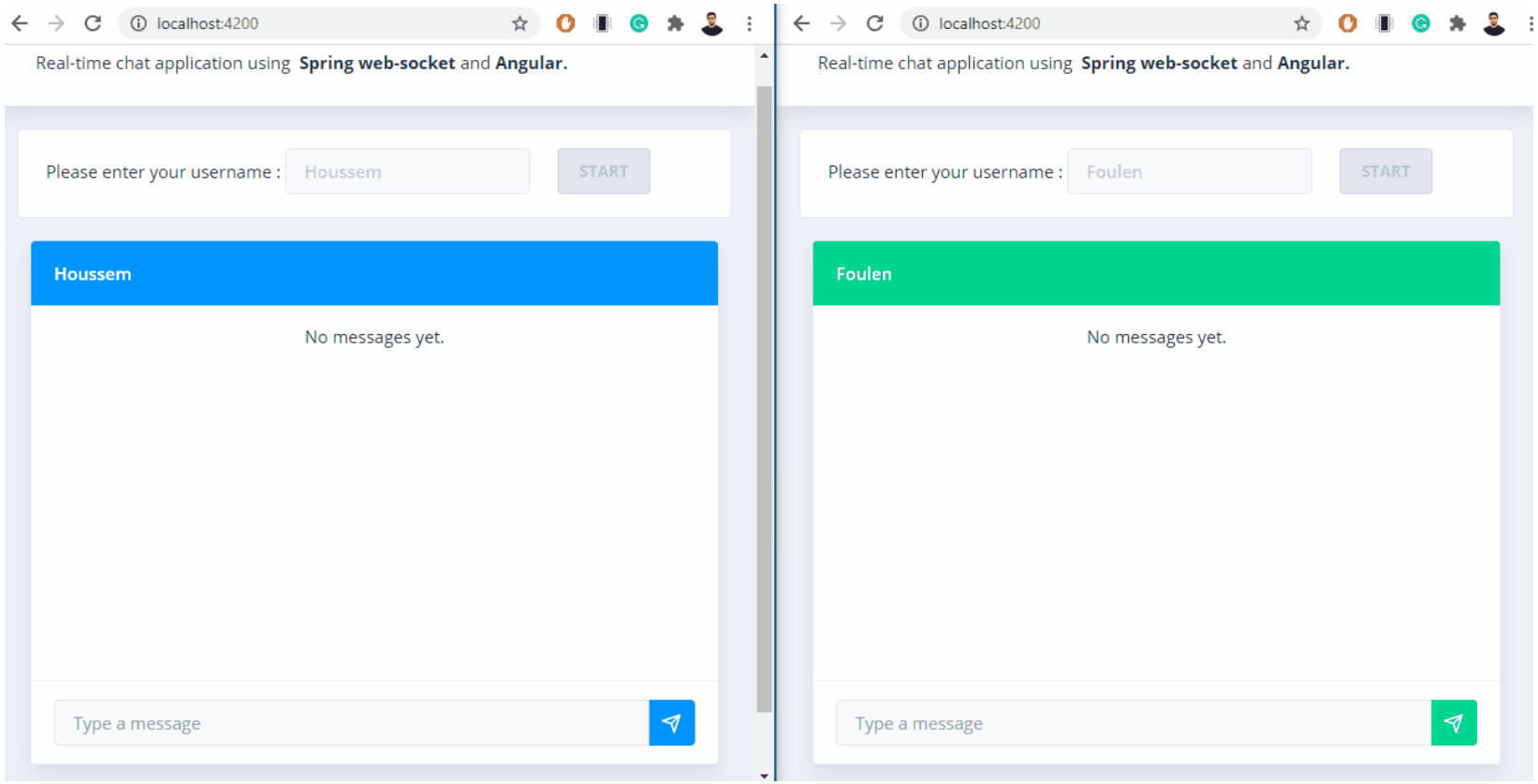
Дизайн приложения

- Работа над дизайном приложения с первого занятия
- Цветовая схема. https://colorscheme.ru



Real-time web

- Ajax
- Push
- WebSocket



Web-фреймворки

- Клиентские фреймворки (Angular, React, Vue)

Предназначены для разработки SPA. Реализуют концепцию «толстого» клиента и «тонкого» сервера. Основная функциональность реализована с использованием JavaScript (и транспилируемых в него языков).

- Серверные фреймворки

Предназначены для разработки приложений на стороне веб-сервера. Реализуют концепцию «тонкого» клиента и «толстого» сервера. Используют традиционные языки веб-разработки: Python, PHP, Ruby, C#, Java, Go ...

Подразделяются на две категории:

- Микрофреймворки (flask)

- Традиционные фреймворки с полной функциональностью (.NET, Spring, Django)

Web разработка на Python

- Интерпретаторы некоторых языков, изначально ориентированных на применение в WWW (например, PHP), обладают встроенным шаблонизатором HTML и могут непосредственно использоваться для веб-разработки.
- В отличие от таких языков, Python для веб-разработки использует исключительно фреймворки.

- Для интеграции с веб-серверами в Python используются спецификация WSGI, которая основана на CGI.

– В частности, для интеграции с веб-сервером Apache разработан модуль Apache mod_wsgi.

– Спецификация WSGI включает такое важное понятие как «Middleware».

- Дальнейшим развитием спецификации WSGI является спецификация ASGI, которая ориентирована на разработку как синхронных, так и асинхронных веб-приложений.

Микрофреймворк Flask

Создание простого приложения:

- Установим виртуальное окружение (windows cmd):

cd <каталог проекта>

python -m venv venv #создадим виртуальное окружение

venv\Scripts\activate #активируем окружение

pip install flask # установим flask

pip list

- Создадим в каталоге проекта Python-файл с простейшим обработчиком URL

- Запустим приложение:

set FLASK_APP=server.py

python -m flask run

- Откроем в браузере адрес - http://127.0.0.1:5000/

Традиционный серверный фреймворк

- Статические файлы (статические HTML-документы, CSS, изображения, сценарии JavaScript и т.д.).
- Контроллеры (обработчики событий пользовательских действий).
- Модели (взаимодействие с БД).
- Представления (view). Шаблоны, генерирующие HTML-страницы и другое динамическое содержимое.

- Конфигурирование фреймворка: действия при запуске приложения, конфигурирование пользовательских сеансов (сессий), переписывание URL (привязка URL к контроллерам), безопасность (аутентификация и авторизация), кэширование, балансировка нагрузки, IOC / DI.

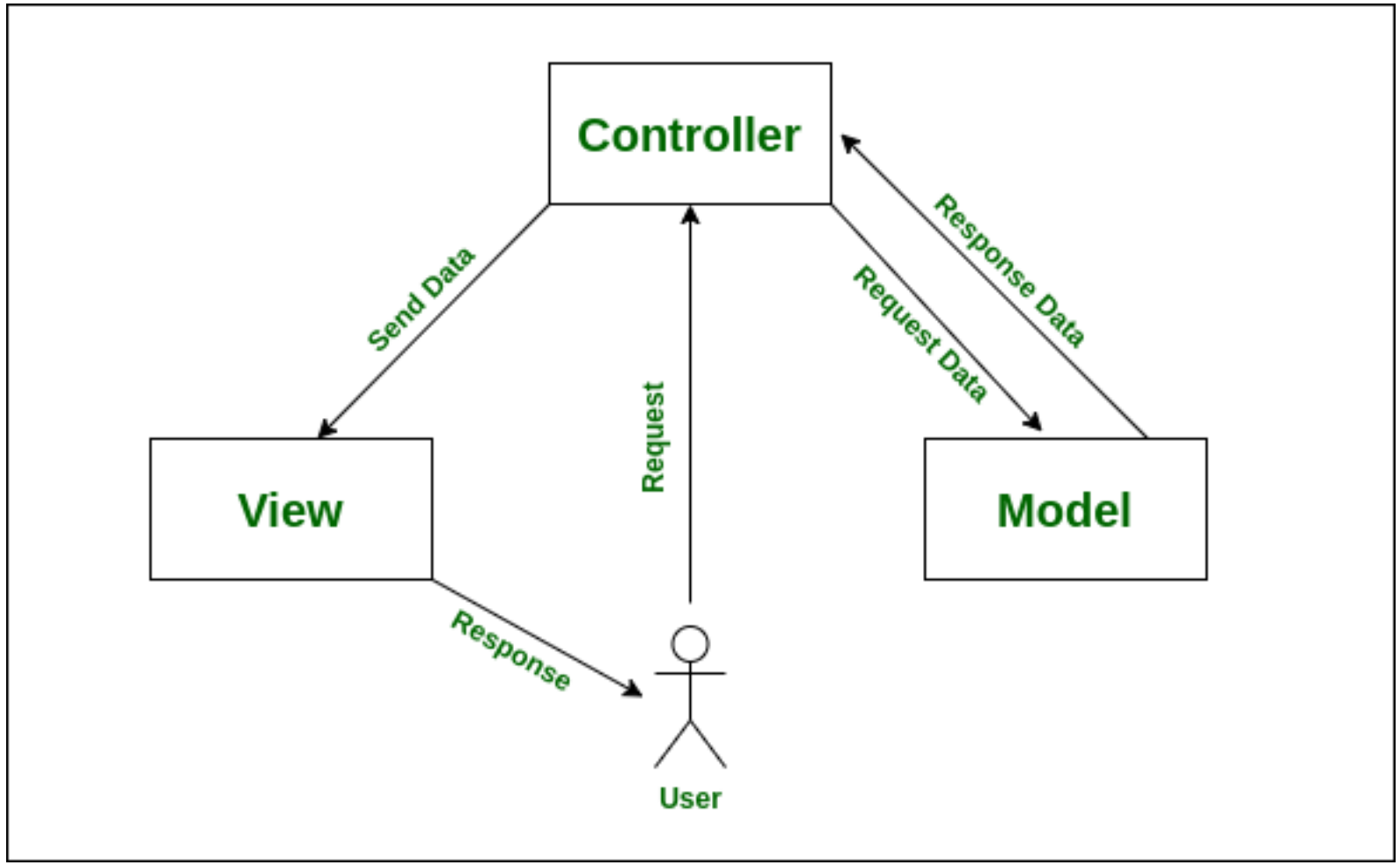
- Утилиты командной строки для управления фреймворком.

– Скаффолдинг (создание структуры проекта, генерация кода контроллеров и представлений на основе моделей, генерация кода приложения на основе специализированных описаний, генерация форм ввода и редактирования данных во время работы приложения).

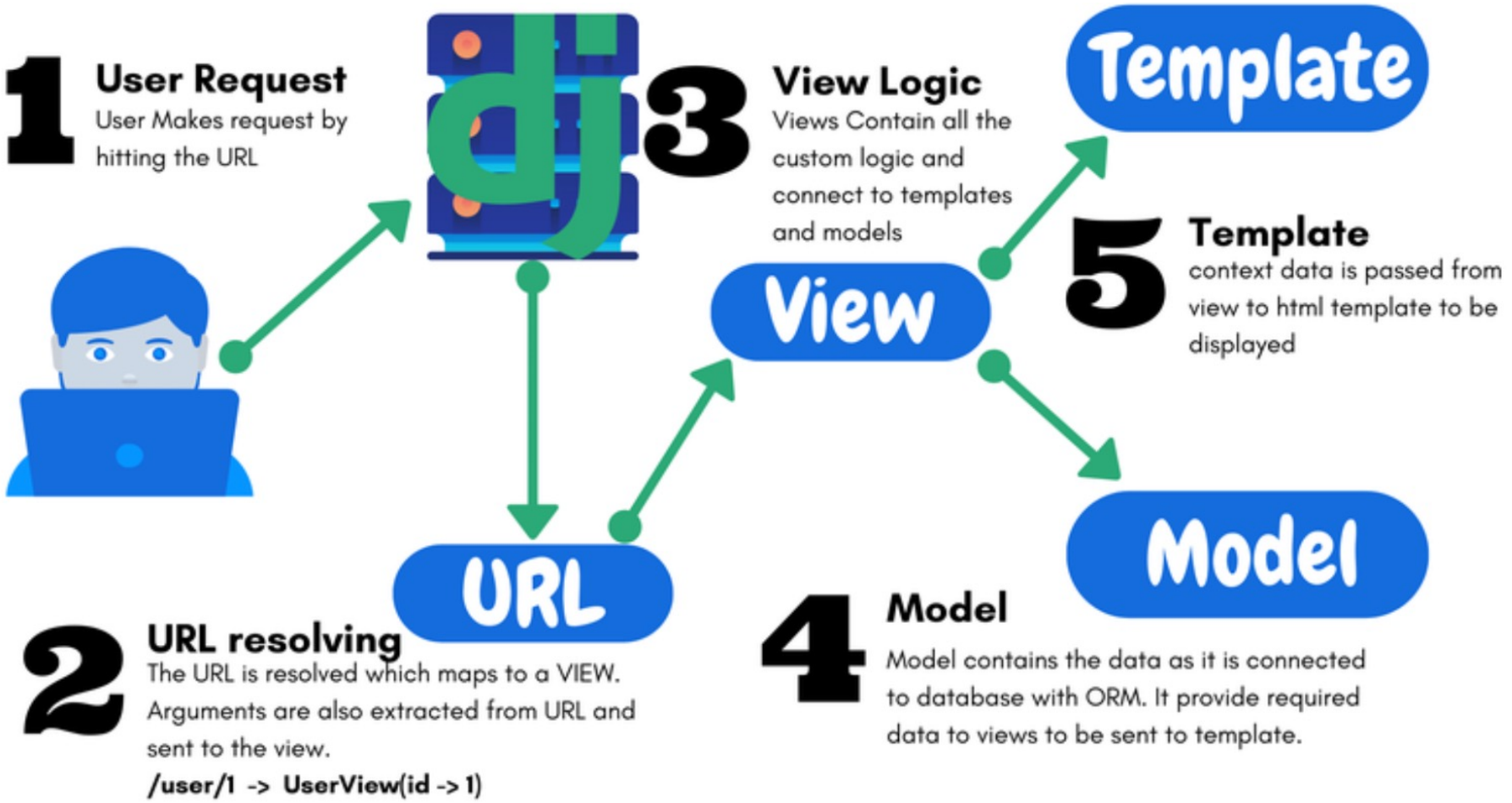
– Миграции (изменение структуры базы данных на основе моделей).

MVC

- Model
- View
- Controller

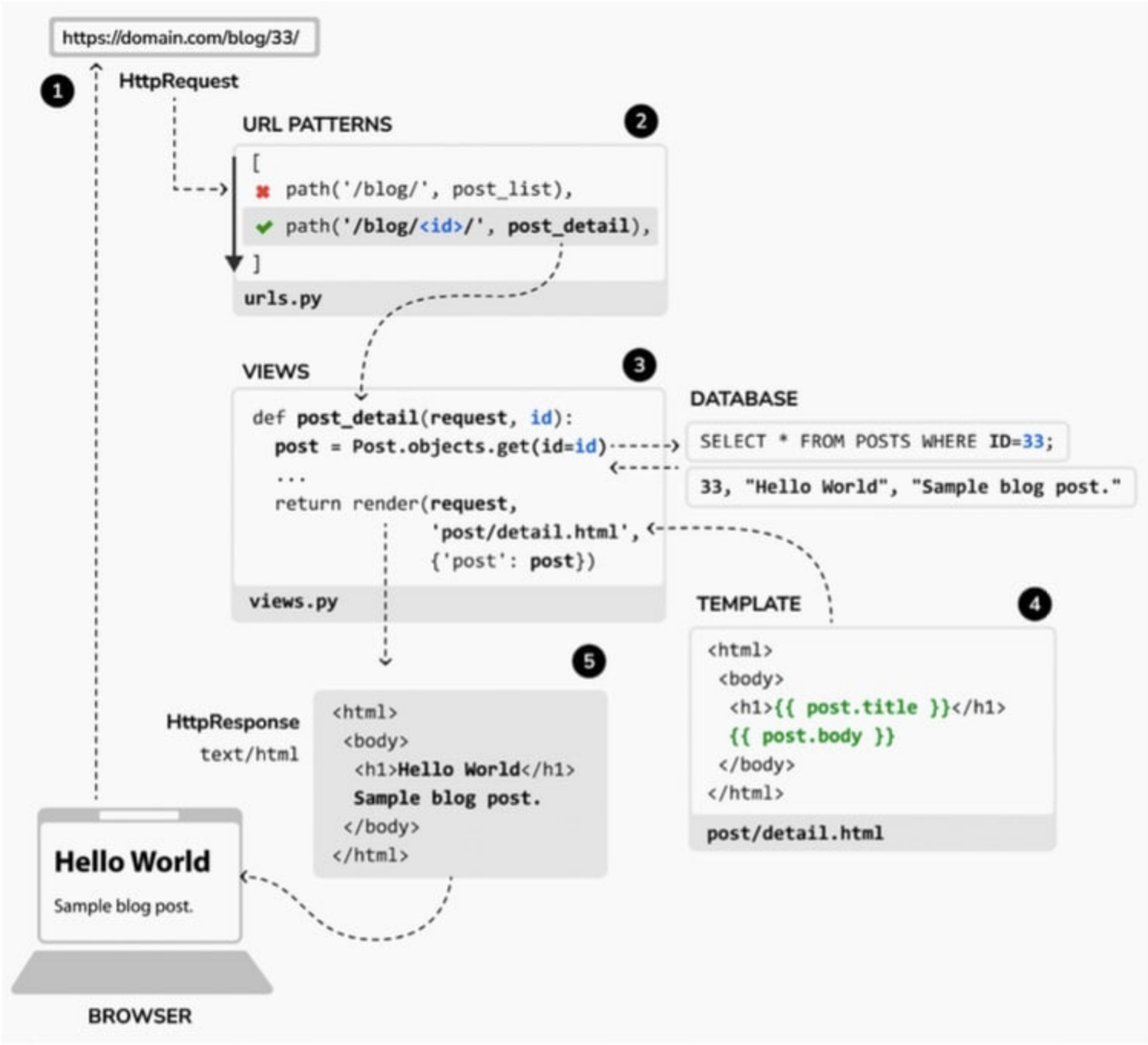


Фреймворк Django. MVC



Django

- Django – это MVC фреймворк
- При обработке запроса сначала обрабатывается URL
- Решается, какой view будет его обрабатывать
- View обращается к БД или нейросети
- Результаты вносятся в шаблон Template, получается HTML



Фреймворк Django. Изучение

- Разделы документации (на русском языке)
  - https://djangodoc.ru/3.2/
  - https://django.fun/docs/django/ru/3.2/
  - https://developer.mozilla.org/ru/docs/Learn/Server-side/Django (учебник из 11 уроков)

- Важные разделы django.fun:
  - Модели (Введение в модели, запросы, миграции)
  - Представления (Обработка URL, представления на основе функций, представления на основе классов, Middleware)
  - Шаблоны (Введение, обзор языка шаблонов)
  - Формы (Введение, формы на основе моделей)
  - Администрирование

REST API

- Django REST https://www.django-rest-framework.org
- Python (3.6, 3.7, 3.8, 3.9, 3.10)
- Django (2.2, 3.0, 3.1, 3.2, 4.0)

- pip install djangorestframework