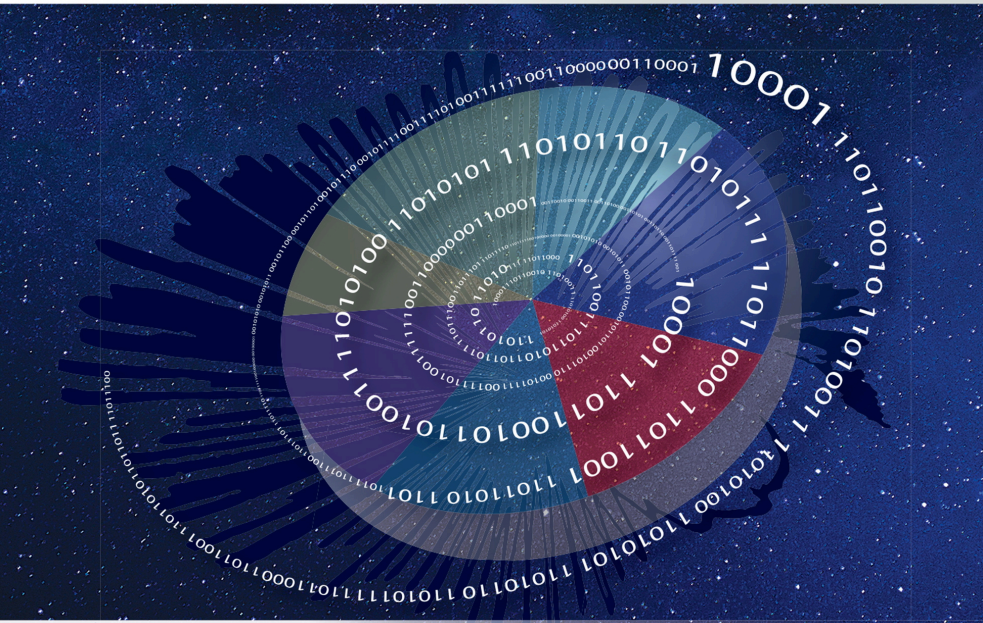
АЙТИШНЫЙ УНИВЕРСИТЕТ

Специалист

А. Н. Баланов

Основы веб-разработки

От интерфейсов до веб-приложений



Агентство электронных изданий
ИНТЕРМЕДИАТОР

УДК 004.9
ББК 32.973я73
Б20

Баланов, Антон Николаевич

Б20 Основы веб-разработки. От интерфейсов до веб-приложений / А. Н. Баланов. — 1 файл pdf : 126 с. — Эл. изд. — Москва : Агентство электронных изданий «Интермедиатор», 2025. — (Айтишный университет. Специалист). — Систем. требования: Adobe Reader XI либо Adobe Digital Editions 4.5 ; экран 10″. — Текст : электронный.

ISBN 978-5-91349-116-9

Книжная серия «Айтишный университет» предназначена для самообразования в сфере информационных технологий, а также может быть использована для формирования учебных курсов в системе профильного высшего образования. Материал излагается сжато и систематично.

Подсерия «Специалист» ориентирована на подготовку профессионалов — программистов-разработчиков, тестировщиков, системных администраторов, веб-мастеров и других специалистов в области информационных технологий.

В настоящей книге в виде краткого, но содержательного обзора, освещены основные вопросы создания веб-приложений, включая описание инструментов разработки, решения задач развертывания, оптимизации, безопасности и др.

В соответствии со ст. 1299 и 1301 ГК РФ при устранении ограничений, установленных техническими средствами защиты авторских прав, правообладатель вправе требовать от нарушителя возмещения убытков или выплаты компенсации.

ISBN 978-5-91349-116-9

© «Интермедиатор», электронное издание, 2025.

ОГЛАВЛЕНИЕ

Глава 1. Введение в веб-разработку 10
Введение 10
Основные принципы и концепции веб-разработки 11
Различие между статическими и динамическими веб-страницами 15
Роль веб-технологий и их влияние на современный Интернет. 17
Заключение 20

Глава 2. Современные технологии и инструменты веб-разработки 22
Введение 22
Обзор популярных языков программирования и фреймворков для веб-разработки 23
Использование интегрированных сред разработки (IDE) и других инструментов для эффективной работы 26
Введение в версионирование кода и системы контроля версий 29
Заключение 32

Глава 3. Процесс создания веб-интерфейсов 34
Введение 34
Основы HTML и CSS для создания веб-страниц 35
Работа с разметкой, стилями и макетами веб-интерфейсов 38
Использование JavaScript и библиотек для создания интерактивных элементов. 41

Заключение 44

Глава 4. Введение в веб-приложения 46
Введение 46
Определение и преимущества веб-приложений 47
Клиент-серверная модель и взаимодействие между клиентом и сервером 49
Разработка архитектуры и основных компонентов веб-приложения 51
Заключение 54

Глава 5. Тестирование в веб-разработке 56
Введение 56
Значение тестирования веб-приложений и его роли в обеспечении качества 57
Виды тестирования веб-приложений. функциональное, нагрузочное, безопасности и т.д. 59
Использование инструментов и фреймворков для автоматического тестирования веб-приложений 62
Заключение 64

Глава 6. Оптимизация производительности веб-приложений 66
Введение 66
Принципы и методы оптимизации производительности веб-приложений 67
Улучшение загрузки и отображения веб-страниц 69
Оптимизация работы клиентской и серверной частей приложения 72
Заключение 76

Глава 7. Работа с базами данных в веб-разработке 77
Введение 77
Основы баз данных и их роль в веб-приложениях 78
Использование языка запросов SQL для управления данными 80

Разработка и оптимизация моделей баз данных для веб-приложений 82
Заключение 86

Глава 8. Развертывание и управление веб-приложениями 87
Введение 87
Планирование и процесс развертывания веб-приложений 88
Управление версиями и обновлениями веб-приложений 90
Мониторинг и обеспечение доступности веб-приложений 94
Заключение 97

Глава 9. Работа с API в веб-разработке 98
Введение 98
Введение в использование API в веб-приложениях 99
Интеграция с внешними сервисами и данными через API 101
Создание и документирование собственного API для веб-приложений 104
Заключение 107

Глава 10. Мобильная веб-разработка 109
Введение 109
Адаптация и оптимизация веб-приложений для мобильных устройств 110
Использование мобильных фреймворков и инструментов для создания мобильных веб-приложений 113
Разработка и тестирование мобильных пользовательских интерфейсов 117
Заключение 119

ГЛАВА 1

ВВЕДЕНИЕ В ВЕБ-РАЗРАБОТКУ

ВВЕДЕНИЕ

Глава 1 посвящена введению в веб-разработку. Мы рассмотрим основные принципы и концепции, которые лежат в основе разработки веб-приложений. Также обсудим различие между статическими и динамическими веб-страницами, а также роль веб-технологий и их влияние на современный Интернет.

Веб-разработка является процессом создания и поддержки веб-приложений, которые доступны через Интернет. Это включает в себя различные аспекты, такие как разработка веб-дизайна, программирование клиентской и серверной частей, управление базами данных и многое другое. Веб-разработчики используют различные языки программирования, фреймворки и инструменты для создания функциональных и эстетически привлекательных веб-сайтов и приложений.

Одним из ключевых аспектов веб-разработки является различие между статическими и динамическими веб-страницами. Статические страницы представляют собой неподвижный контент, который остается неизменным при каждом запросе пользователя. Динамические страницы, напротив, могут генерироваться динамически на сервере в зависимости от ввода пользователя или других факторов. Они позволяют создавать интерактивные и персонализированные веб-приложения.

Развитие веб-технологий имеет существенное влияние на современный Интернет. HTML, CSS и JavaScript являются основными языками веб-разработки, позволяющими создавать структуру, стиль и функциональность веб-страниц. Фреймворки и библиотеки, такие как React, Angular и Vue.js, предоставляют разработчикам мощные инструменты для создания

сложных и масштабируемых веб-приложений. Кроме того, веб-разработка тесно связана с технологиями серверной части, такими как PHP, Python, Ruby и другие.

Введение в веб-разработку предоставляет основы и общую картину процесса создания веб-приложений. Оно поможет вам понять, какие концепции и принципы лежат в основе веб-разработки, и как различные технологии влияют на современный Интернет. Знание этих основных принципов и концепций позволит вам лучше понять последующие главы и более успешно применять веб-технологии при разработке своих проектов.

ОСНОВНЫЕ ПРИНЦИПЫ И КОНЦЕПЦИИ ВЕБ-РАЗРАБОТКИ

Веб-разработка является процессом создания и поддержки веб-приложений и сайтов. Этот процесс включает в себя различные этапы, начиная от планирования и проектирования до реализации и развертывания. Веб-разработчики должны придерживаться определенных принципов и концепций, чтобы создавать эффективные и функциональные веб-приложения.

1. Фронтенд и бэкенд

Веб-разработка обычно включает разработку фронтенда (клиентской части) и бэкенда (серверной части). Фронтенд отвечает за визуальное представление и взаимодействие с пользователем. Он использует языки программирования, такие как HTML, CSS и JavaScript. Бэкенд отвечает за обработку данных, взаимодействие с базами данных и другие серверные операции. Он может быть написан на различных языках, таких как PHP, Python, Ruby, Java и другие.

2. HTML (HyperText Markup Language)

HTML является основным языком разметки веб-страниц и составляет основу веб-разработки. Он определяет структуру и содержимое веб-страницы с помощью различных элементов и тегов. Пример HTML-кода *(см. Рис.1.1)*.

Рисунок 1.1

```
<!DOCTYPE html>
<html>
<head>
  <title>Пример страницы</title>
</head>
<body>
  <h1>Привет, мир!</h1>
  <p>Это пример веб-страницы.</p>
</body>
</html>
```

3. CSS (Cascading Style Sheets)

CSS используется для описания внешнего вида и стиля веб-страниц. Он позволяет управлять цветами, шрифтами, расположением элементов и другими аспектами дизайна. Пример CSS-стилей.

Рисунок 1.2

```
h1 {
  color: blue;
  font-size: 24px;
}

p {
  color: red;
  font-size: 16px;
}
```

4. JavaScript

JavaScript является языком программирования, который позволяет добавлять интерактивность и функциональность

на веб-страницы. С помощью JavaScript можно выполнять различные действия, обрабатывать события и взаимодействовать с пользователем. Пример JavaScript-кода.

Рисунок 1.3

```
function greet() {
  var name = prompt("Введите ваше имя:");
  alert("Привет, " + name + "!");
}
```

Таблица 1.1

Примеры основных языков и технологий веб-разработки

Язык/ технология	***Описание***	***Пример***
HTML	Язык разметки веб-страниц	** Заголовок **
CSS	Язык стилей для веб-страниц	**color: blue;**
JavaScript	Язык программирования	**function greet() { ... }**
PHP	Язык программирования для бэкенда	**echo "Hello, world!";**
Python	Язык программирования для бэкенда	print("Hello, world!")

Основные принципы и концепции веб-разработки являются фундаментом для создания эффективных и качественных веб-приложений. Рассмотрим подробнее эти принципы и концепции.

1. *Клиент-серверная архитектура.* Клиент-серверная архитектура является основой веб-разработки. Веб-приложение

состоит из клиентской части, которая выполняется на стороне браузера пользователя, и серверной части, которая выполняется на сервере. Клиент отправляет запросы на сервер, а сервер обрабатывает эти запросы и отправляет обратно необходимые данные или результаты.

2. *Принцип разделения ответственностей (Separation of Concerns).* Этот принцип заключается в разделении функциональности и ответственностей между различными компонентами веб-приложения. Например, HTML используется для определения структуры содержимого, CSS — для определения внешнего вида и стиля, а JavaScript — для добавления интерактивности и поведения. Разделение этих ответственностей облегчает разработку, поддержку и масштабируемость приложения.

3. *Модульность и повторное использование*. Модульность и повторное использование кода являются важными принципами веб-разработки. Разбиение приложения на отдельные модули и компоненты позволяет легко масштабировать и поддерживать код. Кроме того, повторное использование кода позволяет сократить время разработки и улучшить его качество.

4. *Гибкость и расширяемость*. Веб-приложения должны быть гибкими и легко расширяемыми. Это означает, что приложение должно быть способно адаптироваться к изменяющимся требованиям и масштабироваться по мере необходимости. Гибкость и расширяемость достигаются с помощью правильной архитектуры, использования фреймворков и платформ, а также правильного управления кодом и ресурсами.

5. *Безопасность*. Безопасность является важным аспектом веб-разработки. Веб-приложения должны быть защищены от различных видов атак, таких как инъекции, межсайтовый скриптинг (XSS) и подделка межсайтовых запросов (CSRF). Разработчики должны следовать безопасным практикам, таким как валидация и санитизация пользовательского ввода, использование защищенных протоколов связи и контроль доступа к данным и функциональности.

6. *Оптимизация производительности*. Производительность веб-приложений играет важную роль в пользовательском опыте. Разработчики должны стремиться к оптимизации про-

изводительности, учитывая загрузку и отображение страниц, скорость обработки запросов на сервере, использование кэширования и сжатия данных, а также оптимизацию баз данных и других ресурсов.

7. *Тестирование и отладка*. Тестирование и отладка являются неотъемлемыми частями веб-разработки. Разработчики должны создавать и проводить тесты для проверки функциональности, совместимости и безопасности приложения. Отладка позволяет выявлять и исправлять ошибки и проблемы в коде и настроенных системах.

Основные принципы и концепции веб-разработки обеспечивают основу для создания эффективных и качественных веб-приложений. Понимание этих принципов помогает разработчикам принимать правильные решения при проектировании и разработке веб-приложений, обеспечивая их гибкость, безопасность, производительность и поддерживаемость.

РАЗЛИЧИЕ МЕЖДУ СТАТИЧЕСКИМИ И ДИНАМИЧЕСКИМИ ВЕБ-СТРАНИЦАМИ

Веб-страницы могут быть разделены на две основные категории: статические и динамические. Различие между ними заключается в способе создания и отображения контента. Давайте рассмотрим каждую категорию подробнее.

Статические веб-страницы

Статические веб-страницы являются базовой формой веб-сайта. Они создаются путем написания и сохранения HTML-кода на сервере, а затем отправляются клиенту для отображения веб-страницы в браузере. Контент на статической веб-странице остается неизменным, пока он не будет изменен вручную разработчиком и перезагружен на сервер.

Преимущества статических веб-страниц

1. *Простота создания*. Создание статических веб-страниц не требует специальных навыков программирования или базы данных.

2. *Быстрая загрузка*. Так как статические веб-страницы уже сформированы и сохранены на сервере, их загрузка происходит быстрее, чем динамических страниц.

Недостатки статических веб-страниц

1. *Ограниченная интерактивность*. Статические страницы не могут обеспечить динамический контент или взаимодействие с пользователем.

2. *Трудность в поддержке*. При необходимости внесения изменений на каждой странице сайта требуется ручное редактирование каждой страницы отдельно.

Динамические веб-страницы

Динамические веб-страницы создаются с использованием программных языков, таких как PHP, JavaScript, ASP.NET и других. Контент на динамических страницах генерируется динамически во время выполнения на сервере перед его отправкой клиенту.

Преимущества динамических веб-страниц

1. *Интерактивность*. Динамические страницы позволяют взаимодействие с пользователем, обеспечивая формы обратной связи, поиск, фильтрацию и другие динамические элементы.

2. *Управление содержимым*. Динамические страницы облегчают управление содержимым сайта через административные панели и базы данных.

3. *Повторное использование кода*. Динамические страницы позволяют использовать шаблоны и компоненты для повторного использования кода, что облегчает его поддержку и масштабирование.

Недостатки динамических веб-страниц

1. *Сложность разработки*. Создание динамических веб-страниц требует знания программирования и работы с базами данных.

2. *Замедление загрузки.* Динамические страницы требуют времени на генерацию контента на сервере, что может замедлить загрузку страницы.

Вот небольшая таблица, иллюстрирующая различия между статическими и динамическими веб-страницами.

Таблица 1.2

	Статические веб-страницы	***Динамические веб-страницы***
Создание	Простое	Требует программирования
Интерактивность	Ограниченная	Высокая
Управление содержимым	Труднее	Удобнее
Повторное использование кода	Ограниченное	Легко
Загрузка	Быстрая	Может быть медленнее

РОЛЬ ВЕБ-ТЕХНОЛОГИЙ И ИХ ВЛИЯНИЕ НА СОВРЕМЕННЫЙ ИНТЕРНЕТ

Веб-технологии играют ключевую роль в развитии и функционировании современного Интернета. Они обеспечивают инфраструктуру и инструменты, необходимые для создания, развертывания и взаимодействия с веб-приложениями и сайтами. В этом тексте мы рассмотрим несколько важных веб-технологий, их влияние на современный Интернет, а также предоставим примеры и таблицы для более наглядного представления.

1. HTML (Hypertext Markup Language)

HTML является основным языком разметки веб-страниц. Он определяет структуру и содержимое веб-документа. HTML используется для создания элементов, таких как заголовки, параграфы, списки, таблицы и изображения. Благодаря HTML мы