Кейс-задача № 1

Вопрос 1. Рассмотрите и опишите какие языки программирования необходимо знать для работы в сфере веб-разработки?

Для разработки веб-сайтов и интерактивных веб-приложений используются специальные языки программирования. Они позволяют создавать контент, оформление и обеспечивать работоспособность ресурсов в интернете. Эти языки применяются как для разработки клиентской части (то, с чем взаимодействует пользователь), так и для серверной части (логика работы сервера и управления базами данных).

Несмотря на существование более чем семи тысяч языков программирования, наибольшее распространение в веб-разработке получили HTML, CSS, Python, JavaScript, Java, C++, C#, PHP, Perl, GoLang и Ruby.

Программист выбирает язык, наиболее подходящий для решения конкретных задач. Однако, в большинстве своем, языки программирования имеют схожую структуру, что позволяет понимать общий смысл кода.

Веб-разработку принято разделять на два основных направления: фронтенд и бэкенд. Фронтенд отвечает за разработку пользовательского интерфейса, с которым непосредственно взаимодействуют посетители сайта. Бэкенд включает в себя внутреннюю логику сайта, обработку данных и взаимодействие с базами данных. Фронтенд отправляет запросы на бэкенд, который, в свою очередь, обрабатывает их, обращается к серверу и базам данных, а затем возвращает результаты на фронтенд. Выбор конкретного языка программирования и других технологий зависит от того, с какой частью веб-приложения разработчик предпочитает работать.

Языки для фронтенд-разработчиков

При создании пользовательского интерфейса веб-сайтов используются три основные технологии, однако языком программирования среди них считается только JavaScript.

1. HTML

HTML – это язык разметки гипертекста. Изначально, когда интернет-ресурсы были простыми и представляли собой обычные веб-страницы, он применялся для создания сайтов. Сегодня HTML по-прежнему является основой веб-страниц, определяя расположение элементов, таких как таблицы, изображения и текстовая информация.

HTML состоит из тегов, представляющих собой команды для браузера, указывающие, как отображать содержимое. При вводе адреса сайта браузер отправляет запрос и получает HTML-файл в ответ. Затем браузер преобразует теги в визуальное представление страницы. HTML – это основа любого веб-документа и азбука для веб-разработчика, знакомая как фронтенд-, так и бэкенд-разработчикам.

1. CSS

CSS (каскадные таблицы стилей) – это вторая ключевая технология в современной веб-разработке. Если HTML – это структура сайта, то CSS отвечает за его внешний вид. Сайты, созданные только на HTML, обладают структурой, но выглядят невыразительно. CSS позволяет улучшить оформление, делая сайт привлекательным и удобным. CSS подключается к HTML-основе для задания единых стандартов внешнего вида, таких как размер заголовков, цвета и размеры кнопок. Хотя некоторые параметры оформления можно задать с помощью HTML-тегов, CSS упрощает этот процесс, позволяя создавать стандарты оформления один раз и применять их ко всем страницам.

1. JavaScript

JavaScript – самый популярный язык программирования для создания интерактивных сайтов, реагирующих на действия пользователя. До его появления для отображения изменений требовалась перезагрузка страницы.

JavaScript – это прорыв в веб-разработке. Он работает на основе скриптов, выполняемых браузером. Например, нажатие кнопки может активировать всплывающее окно, данные из которого обработаются на стороне клиента без запросов к серверу. Это уменьшает время ожидания и снижает нагрузку на трафик и сервер. Кнопки, формы, меню, автозаполнение – все это реализуется на JavaScript. Его также используют для бэкенда, но реже. JavaScript считается относительно простым в изучении

Для JavaScript разработано множество фреймворков и библиотек. Знание одного из них – часто обязательное условие при трудоустройстве. Самые популярные: Angular, React и Vue.

Языки для бэкенд-разработчиков

1. PHP

PHP, разработанный в 1995 году, остается одним из самых зрелых и востребованных языков в веб-разработке. Около 76% веб-сайтов в сети используют его, что свидетельствует о его широком распространении. Это лидерство во многом объясняется наличием готовых систем управления контентом (CMS), таких как WordPress. Согласно индексу TIOBE на март 2024 года, PHP занимает 12-е место, а по данным GitHub – 6-е место среди всех языков программирования и 3-е место среди серверных технологий.

В настоящее время PHP позволяет разрабатывать веб-сайты различной сложности, от простых страниц, созданных с помощью готовых инструментов, до масштабных платформ, таких как Wikipedia, BlaBlaCar, Facebook или «ВКонтакте».

Своей популярностью PHP обязан простоте использования и совместимости с различными технологиями, включая распространенные базы данных и веб-серверы. Язык активно развивается, а последняя версия 8.3 была выпущена в 2023 году. Кроме того, у PHP обширное сообщество, которое поддерживает и продвигает технологию. PHP – это специализированный язык, ориентированный исключительно на веб-разработку. В этой области он выделяется своей доступностью, находя применение как в крупных проектах, так и при создании небольших коммерческих веб-сайтов.

Среди наиболее популярных фреймворков для веб-разработки на PHP можно выделить Laravel и Symfony.

PHP, подобно JavaScript, является языком с динамической типизацией. Это может снижать надежность, но упрощает процесс обучения, что делает его одним из самых доступных для начинающих разработчиков.

1. Python

В отличие от PHP, Python выделяется своей универсальностью как язык программирования. Его сфера применения простирается далеко за пределы веб-разработки, охватывая такие области, как анализ данных, машинное обучение, разработка программ для настольных компьютеров и даже создание игр. Согласно индексу TIOBE за март 2024 года, Python лидирует по популярности. В исследовании GitHub он занимает третье место среди всех языков и четвертое среди серверных технологий.

В веб-разработке Python используется для создания серверных приложений. Примером может служить Google, где на Python разработаны отдельные компоненты поисковой системы. Кроме того, серверные части Dropbox и YouTube также функционируют на Python.

Python – это интерпретируемый язык, что означает отсутствие необходимости в компиляции перед запуском. Код можно запускать на любом компьютере, где установлен интерпретатор. Благодаря динамической типизации, код на Python получается компактным и выразительным.

Популярность Python облегчает процесс обучения, так как большинство проблем и ошибок можно оперативно найти в сети. Обширное сообщество готово оказать поддержку начинающим. Дополнительным преимуществом является многофункциональность Python, позволяющая веб-разработчикам легко переключаться между проектами или даже сферами деятельности, например, от разработки серверных приложений к машинному обучению.

Python располагает широким выбором фреймворков и библиотек. Веб-разработчики обычно отдают предпочтение Django, Pyramid или Flask.

1. Java

Java является востребованным языком программирования, особенно в крупных организациях, таких как финансовые учреждения и банки. Он часто используется для разработки серверной части высокопроизводительных приложений, где критически важна безопасность. Согласно индексу TIOBE за март 2024 года, Java занимает четвертое место по популярности среди языков программирования. Исследование GitHub показывает, что он находится на втором месте среди всех языков и лидирует в сфере бэкенд-разработки.

Такие компании, как Amazon, Goldman Sachs и Netflix, используют Java для своих серверных приложений.

Язык был разработан компанией Oracle в 1995 году. Java изначально задумывался как более удобная альтернатива языкам C и C++, и в результате стал столь же надежным, но более простым в использовании.

Стоит отметить, что изучение Java может быть сложнее, чем освоение других популярных бэкенд-языков, таких как Python или PHP. Однако это компенсируется повышенной надежностью разрабатываемого кода.

Java — это компилируемый язык. Программы, написанные на Java, сначала преобразуются в байт-код, который затем выполняется виртуальной машиной. Это обеспечивает высокую скорость работы и повышает безопасность. Компилятор проверяет код на наличие ошибок и соответствие стандартам. Одной из особенностей Java является многословность, что означает, что код может занимать больше строк, а синтаксис более строгий и сложный. Входной порог для изучения этого языка несколько выше, чем у аналогов.

Одним из самых популярных и мощных фреймворков для Java является Spring.

1. Node.js

Node.js представляет собой специализированную платформу, обеспечивающую выполнение JavaScript-кода на серверной стороне. Это означает, что с её помощью можно разрабатывать серверные приложения, используя этот язык программирования. Node.js появился относительно недавно, в 2008 году, и, несмотря на это, его востребованность не столь высока, как у более ранних языков. Согласно исследованиям GitHub, он занимает седьмую позицию по популярности среди языков, используемых для бэкенд-разработки. Однако, в сфере full-stack разработки эта технология занимает второе место по популярности.

Причина кроется в удобстве использования одного и того же языка программирования для создания как фронтенда, так и бэкенда. Это позволяет снизить издержки на разработку, оптимизировать размер команды и обеспечить более тесную интеграцию между используемыми технологиями. Node.js привлекает разработчиков, стремящихся к универсальности. Освоение JavaScript и Node.js представляется более простым, чем изучение JavaScript в сочетании с другим, независимым языком программирования для бэкенда. Следовательно, в перспективе его востребованность будет только расти. В настоящее время Node.js применяется в продуктах таких компаний, как Netflix, Uber и eBay.

1. Ruby

Хотя Ruby является универсальным языком программирования, наибольшее распространение он получил в сфере веб-разработки. Разработанный в 1995 году с намерением создать простой и удобный в использовании язык, Ruby приобрел широкую известность лишь в 2005 году, когда появился фреймворк Ruby on Rails, специально разработанный для веб-приложений. Согласно индексу TIOBE на март 2024 года, Ruby занимает восемнадцатую позицию по популярности. В исследовании GitHub язык занимает двенадцатое место среди всех языков программирования и восьмое место среди технологий, используемых для серверной разработки.

Ruby характеризуется своим кратким синтаксисом и динамической типизацией. Этот язык программирования находит применение в электронной коммерции и при создании стартапов, поскольку позволяет оперативно реализовывать сложные решения. Как правило, его используют в сочетании с фреймворком Ruby on Rails.

1. SQL

SQL – это специализированный язык, используемый для работы с базами данных, а не полноценный язык программирования. Его основная цель – управление данными, поэтому разработка приложений или веб-сайтов на нем невозможна. Однако, SQL является важным инструментом для любого разработчика серверной части (backend), и знание его основ необходимо.

Современные приложения хранят информацию в базах данных, представляющих собой большие таблицы с упорядоченными ячейками. Для навигации и манипулирования этими данными используется SQL. С его помощью можно извлекать, обновлять, сортировать данные, а также объединять различные таблицы. SQL позволяет создавать интерактивные запросы, соответствующие общепринятым стандартам, и совместим с большинством популярных СУБД. Благодаря своей простой структуре, SQL относительно легко изучить.

Вопрос 2. Какие принципы и паттерны программирования широко используются при создании веб-приложений?

При создании веб-приложений широко используются принципы проектирования, паттерны и технологии, связанные с веб-разработкой. Эти элементы помогают создавать интерактивные функции и сервисы, которые изменяются в зависимости от действий пользователя.

Принципы, которые применяются при проектировании веб-приложений:

- Принцип единственной ответственности – каждый компонент должен иметь только одну ответственность.

- Принцип открытости/закрытости – код должен быть открыт для расширения (если необходимо внести изменения, они должны быть выполнены без изменения существующего кода.

- Принцип разделения интерфейсов – приложение должно быть разделено на логические части, которые не зависят друг от друга.

- Принцип инверсии зависимостей – зависимости между компонентами должны быть определены на уровне абстракции, а не на уровне реализации.

- Принцип композиции – компоненты должны быть построены из более мелких компонентов, что позволяет легко добавлять, изменять или удалять функциональность.

Паттерны программирования — это особые способы или решения, которые программисты используют для решения обычных задач при написании программ. Эти способы помогают сделать код более понятным, удобным для работы и повторного использования. Паттерны - это своего рода шаблоны, которые предоставляют соглашения о том, как лучше всего структурировать и организовать код для достижения нужных целей.

1. Singleton (Одиночка)

Паттерн *Singleton* предназначен для контроля над созданием экземпляров класса, гарантируя, что в приложении существует только один его экземпляр. Достигается это за счет приватного конструктора, который блокирует возможность создания объектов класса напрямую извне. Внутри класса объявляется статическая переменная, которая и хранит этот единственный объект. Для получения доступа к экземпляру используется статический метод, который создает экземпляр при первом обращении или возвращает существующий при последующих.

Такой подход предоставляет глобальную точку доступа к одному и тому же объекту, что особенно ценно при работе с общими ресурсами, параметрами конфигурации или при ведении логов. Singleton гарантирует единичность экземпляра, что повышает эффективность использования ресурсов и облегчает поддержку кодовой базы.

В разработке программного обеспечения Singleton полезен в ситуациях, когда необходим централизованный контроль доступа к ресурсам, таким как подключение к базе данных, параметры приложения или система протоколирования.

Примеры использования Singleton:

*Настройки приложения*: Хранение и доступ к общим настройкам и конфигурации приложения.

*База данных*: Управление единственным соединением с базой данных.

*Логирование*: Централизованное логирование для всего приложения.

*Управление ресурсами*: Создание и управление ресурсами, такими как пулы потоков или соединений.

*Кэширование данных*: Хранение часто используемых данных для оптимизации производительности.

*Глобальный доступ*: Доступ к общему объекту из разных частей приложения.

*Счетчики и идентификаторы*: Создание уникальных счетчиков или идентификаторов.

*Реализация паттернов других паттернов*: например, используется в фабричных методах или строителях для гарантии единственного создания объектов.

1. Factory Method (Фабричный метод)

Фабричный метод – это паттерн проектирования, относящийся к порождающим паттернам. Он определяет общий интерфейс для конструирования объектов в родительском классе, перекладывая выбор конкретного класса создаваемого объекта на дочерние классы. Это позволяет перенести ответственность за создание экземпляров на подклассы, обеспечивая более адаптивную и масштабируемую структуру приложения.

Механизм работы фабричного метода:

1. Определяется абстрактный родительский класс (или интерфейс), содержащий метод (фабричный метод), предназначенный для создания объекта.
2. Дочерние классы реализуют этот фабричный метод, возвращая определенные экземпляры объектов, соответствующие их специфике или потребностям.
3. Клиентский код взаимодействует с абстрактным родительским классом и вызывает фабричный метод для получения объектов, не вдаваясь в детали их конкретной реализации.

Примеры использования:

*Драйверы устройств*: Драйверы устройств в операционных системах используют Фабричные методы для создания экземпляров устройств, таких как принтеры или сканеры.

*Создание объектов в играх*: В игровой разработке Фабричные методы могут использоваться для создания персонажей, оружия, монстров и других игровых объектов.

*Подключение к базам данных*: Библиотеки для работы с базами данных могут использовать Фабричные методы для создания соединений с разными типами баз данных.

*Фреймворки и библиотеки*: Фабричные методы часто применяются в фреймворках и библиотеках, чтобы предоставить клиентам способ создания пользовательских объектов, не изменяя основной код фреймворка.

**3**. Observer (Наблюдатель)

Паттерн *Observer* предназначен для установления связи типа "один ко многим" между компонентами системы. Это означает, что когда состояние одного объекта претерпевает изменения, все объекты, зависящие от него, получают автоматическое уведомление и выполняют обновление.

Такой подход обеспечивает взаимодействие и синхронизацию между элементами системы, избегая жесткой взаимозависимости между ними. Другими словами, наблюдаемые объекты не имеют прямого знания о наблюдателях.

Благодаря этому, система становится более гибкой и расширяемой, так как добавление или удаление наблюдателей не требует внесения изменений в наблюдаемый объект. Это способствует упрощению процесса разработки и поддержки программного обеспечения.

Использование на практике:

Паттерн Observer используется в различных областях программирования и приложений, включая: графические интерфейсы пользователя (GUI), где виджеты могут быть наблюдателями за моделью данных; реализация шаблона "издатель-подписчик" в системах сообщений и событий; очереди событий и обработчики событий; реактивное программирование и библиотеки, такие как RxJava и RxSwift.

**4**. Паттерн Strategy (Стратегия)

Паттерн Strategy (Стратегия) позволяет определить семейство алгоритмов, инкапсулировать каждый из них и сделать их взаимозаменяемыми. Это позволяет выбирать конкретный алгоритм во время выполнения программы без изменения кода клиента, что делает код более гибким и обеспечивает его расширяемость.

Использование на практике:

*Системы сортировки*: Вы можете использовать паттерн Стратегия для реализации разных алгоритмов сортировки (например, сортировка пузырьком, сортировка слиянием, быстрая сортировка) и переключать между ними в зависимости от требований приложения.

*Парсинг данных*: если у вас есть разные форматы данных (например, JSON, XML, CSV), вы можете использовать Стратегию для выбора соответствующего парсера в зависимости от формата данных.

*Авторизация и аутентификация*: В зависимости от требований безопасности, вы можете использовать Стратегию для выбора разных методов аутентификации, таких как логин-пароль, OAuth, JWT.

*Генерация отчетов*: если у вас есть разные форматы отчетов (PDF, Excel, HTML), вы можете использовать Стратегию для генерации отчетов в выбранном формате.

**5**. MVC (Model-View-Controller)

Паттерн MVC (Model-View-Controller) разделяет приложение на три основных компонента

*Модель (Model)*: Этот компонент представляет данные и бизнес-логику приложения. Модель отвечает за хранение, обработку и обновление данных. Она не зависит от представления и контроллера.

*Представление (View)*: Представление отображает данные из модели пользователю. Это компонент, который отвечает за отображение информации и интерфейс взаимодействия с пользователем. Представление не содержит бизнес-логику и не взаимодействует напрямую с моделью.

*Контроллер (Controller)*: Контроллер обрабатывает пользовательский ввод и управляет моделью и представлением. Он принимает запросы от пользователя через представление, взаимодействует с моделью для получения или обновления данных и обновляет представление с учетом изменений в модели.

Использование на практике:

*Веб-приложения*: MVC широко используется в веб-разработке для разделения логики бэкенда (модель), отображения веб-страниц (представление) и обработки HTTP-запросов (контроллер).

*Десктопные приложения*: MVC может быть использован в десктопных приложениях для создания графического пользовательского интерфейса, где модель представляет данные, представление отображает их и контроллер обрабатывает действия пользователя.

*Мобильные приложения*: аналогично веб- и десктоп-приложениям, MVC может быть применен в мобильных приложениях для обеспечения разделения между бизнес-логикой, пользовательским интерфейсом и обработкой событий.