Объе?ктно-ориенти?рованное программи?рование (ООП) — методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного класса[1], а классы образуют иерархию наследования.

ООП представляет программное обеспечение как совокупность взаимодействующих объектов

Абстракция данных

Абстрагирование означает выделение значимой информации и исключение из рассмотрения незначимой. В ООП рассматривают лишь абстракцию данных (нередко называя её просто «абстракцией»), подразумевая набор значимых характеристик объекта, доступный остальной программе.

Абстракция

Совокупность комплексных наследований, методов и свойств объекта должны адекватно отражать модель реальности.

Инкапсуляция[5]

Инкапсуляция — свойство системы, позволяющее объединить данные и методы, работающие с ними, в классе. Одни языки (например, С++, Java или Ruby) отождествляют инкапсуляцию с сокрытием, но другие (Smalltalk, Eiffel, OCaml) различают эти понятия.

В информатике сокрытие представляет собой принцип проектирования, заключающийся в разграничении доступа различных частей программы к внутренним компонентам друг друга

Наследование[5]

Наследование — свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью. Класс, от которого производится наследование, называется базовым, родительским или суперклассом. Новый класс — потомком, наследником, дочерним или производным классом.

Полиморфизм подтипов[5]

Полиморфизм подтипов (в ООП называемый просто «полиморфизмом») — свойство системы, позволяющее использовать объекты с одинаковым интерфейсом без информации о типе и внутренней структуре объекта. .

Класс

Определяет характеристики объекта. Класс является описанием шаблона свойств и методов объекта.

Объект

Сущность в адресном пространстве вычислительной системы, появляющаяся при создании экземпляра класса (например, после запуска результатов компиляции и связывания исходного кода на выполнение).

Замыкание – это функция вместе со всеми внешними переменными, которые ей доступны.

«Полифилл» (англ. polyfill) – это библиотека, которая добавляет в старые браузеры поддержку возможностей, которые в современных браузерах являются встроенными.

Лексическое окружение

Все переменные внутри функции – это свойства специального внутреннего объекта LexicalEnvironment, который создаётся при её запуске.

Мы будем называть этот объект «лексическое окружение» или просто «объект переменных».

Дескриптор – объект, который описывает поведение свойства.

Декоратор – приём программирования, который позволяет взять существующую функцию и изменить/расширить ее поведение.

Значение this называется контекстом вызова и будет определено в момент вызова функции.

AJAX (аббревиатура от «Asynchronous Javascript And Xml») – технология обращения к серверу без перезагрузки страницы.

COMET – общий термин, описывающий различные техники получения данных по инициативе сервера.

Делегирование – это средство оптимизации интерфейса. Мы используем один обработчик для схожих действий на однотипных элементах.

Можно сказать, что AJAX – это «отправил запрос – получил результат», а COMET – это «непрерывный канал, по которому приходят данные».

Протокол WebSocket предназначен для решения любых задач и снятия ограничений обмена данными между браузером и сервером.

Он позволяет пересылать любые данные, на любой домен, безопасно и почти без лишнего сетевого трафика.

HTTP (англ. HyperText Transfer Protocol — «протокол передачи гипертекста») — протокол прикладного уровня передачи данных (изначально — в виде гипертекстовых документов в формате «HTML», в настоящий момент используется для передачи произвольных данных). Основой HTTP является технология «клиент-сервер», то есть предполагается существование:

• Потребителей (клиентов), которые инициируют соединение и посылают запрос;

• Поставщиков (серверов), которые ожидают соединения для получения запроса, производят необходимые действия и возвращают обратно сообщение с результатом.

HTTP используется также в качестве «транспорта» для других протоколов прикладного уровня, таких как SOAP, XML-RPC, WebDAV.

HTTP — протокол прикладного уровня; аналогичными ему являются FTP и SMTP. Обмен сообщениями идёт по обыкновенной схеме «запрос-ответ»

IFRAME -- это окно браузера, вложенное в основное

--------------------------

В этой главе мы рассмотрим работу с форматом JSON, который используется для представления объектов в виде строки.

Это один из наиболее удобных форматов данных при взаимодействии с JavaScript. Если нужно с сервера взять объект с данными и передать на клиенте, то в качестве промежуточного формата – для передачи по сети, почти всегда используют именно его.

Протокол JSONP

Если создать тег <script src>, то при добавлении в документ запустится процесс загрузки src. В ответ сервер может прислать скрипт, содержащий нужные данные.

Таким образом можно запрашивать данные с любого сервера, в любом браузере, без каких-либо разрешений и дополнительных проверок.

Протокол JSONP – это «надстройка» над таким способом коммуникации.

JSONP или «JSON with padding» (JSON с набивкой) — это дополнение к базовому формату JSON. Он предоставляет способ запросить данные с сервера, находящегося в другом домене — операцию, запрещённую в типичных веб-браузерах из-за политики ограничения домена.

Оператор instanceof позволяет проверить, какому классу принадлежит объект, с учетом прототипного наследования.

Promise (обычно их так и называют «промисы») – предоставляют удобный способ организации асинхронного кода. В современном JavaScript промисы часто используются в том числе и неявно, при помощи генераторов, но об этом чуть позже.

Promise – это специальный объект, который содержит своё состояние. Вначале pending («ожидание»), затем – одно из: fulfilled («выполнено успешно») или rejected («выполнено с ошибкой»).

===================================

Модулем считается файл с кодом.

В этом файле ключевым словом export помечаются переменные и функции, которые могут быть использованы снаружи.

Другие модули могут подключать их через вызов import.

===================================

Прокси (proxy) – особый объект, смысл которого – перехватывать обращения к другому объекту и, при необходимости, модифицировать их.

Генераторы – новый вид функций в современном JavaScript. Они отличаются от обычных тем, что могут приостанавливать своё выполнение, возвращать промежуточный результат и далее возобновлять его позже, в произвольный момент времени.

use strict – используется для того, чтобы перевести код в режим полного соответствия современному стандарту.

TRY …. CATCH

Таким образом, при ошибке в try скрипт не «падает», и мы получаем возможность обработать ошибку внутри catch.

Деструктуризация (destructuring assignment) – это особый синтаксис присваивания, при котором можно присвоить массив или объект сразу нескольким переменным, разбив его на части.

Итераторы

В современный JavaScript добавлена новая концепция «итерируемых» (iterable) объектов.

Итерируемые или, иными словами, «перебираемые» объекты – это те, содержимое которых можно перебрать в цикле.

Оператор spread (троеточие) в объявлении позволяет функции получать оставшиеся аргументы в массив

DNS (англ. Domain Name System — система доменных имён) — компьютерная распределённая система для получения информации о доменах. Чаще всего используется для получения IP-адреса по имени хоста (компьютера или устройства), получения информации о маршрутизации почты, обслуживающих узлах для протоколов в домене (SRV-запись).

Распределённая база данных DNS поддерживается с помощью иерархии DNS-серверов, взаимодействующих по определённому протоколу.

Со?кеты (англ. socket — разъём) — название программного интерфейса для обеспечения обмена данными между процессами. Процессы при таком обмене могут исполняться как на одной ЭВМ, так и на различных ЭВМ, связанных между собой сетью. Сокет — абстрактный объект, представляющий конечную точку соединения.

Следует различать клиентские и серверные сокеты. Клиентские сокеты грубо можно сравнить с конечными аппаратами телефонной сети, а серверные — с коммутаторами. Клиентское приложение (например, браузер) использует только клиентские сокеты, а серверное (например, веб-сервер, которому браузер посылает запросы) — как клиентские, так и серверные сокеты.