1. **Перечислите основные свойства глобальных объектов Node.js и поясните их предназначение.**

Основные глобальные объекты: global, process, buffer.

Все они доступны из любых модулей, т.е. из вообще любого места в программе без импортирования.

*Global:* Некоторые из объектов global: require, module, exports, \_\_filename, \_\_dirname.

Смысл в том, что переменные\объекты не являются полностью глобальными, а имеют видимость на уровне модуля. То есть в разных модулях можно объявить две переменные с одним именем без конфликта. Чтобы переменная\объект была видна вообще во всем приложении и всех модулях, надо объявить ее через global.varName = “value”

*Process:* Соответствует объекту процесса, который выделяется для программы. Имеет много методов для понимания того, что происходит в процессе. Есть методы для ввода\вывода в командную строку: stdin, stdout, stderr, и метод для корректного завершения приложения: exit().Также хранит инфу о среде выполнения программы (node -p "process.versions" / “process.env” / “process.release”).

Основные события process:

* beforeExit – перед тем, как в Event Loop больше не остается событий
* exit – идет после beforeExit; в общем случае процесс завершается, когда события в коде кончились; однако листенер на событие beforeExit может выполнять асинхронные вызовы. Вызовите process.exit() в любом месте программы, чтобы завершить процесс и выйти из приложения.

*Buffer:* Нужен для обработки двоичных данных. Каждый буфер соответствует некоторой памяти. Похожи на массивы целых чисел, но нельзя ресайзнуть. Можно выделить n бит под буфер и записывать туда бинарные данные.

1. **Поясните понятие «асинхронная функция».**

В асинхронной функции поток не будет блокироваться, будут отдаваться необходимые задачи, а по их завершению будет вызвана коллбэк-функция. Работа происходит в 2 фазы: отправка запроса и получение результата. То есть, процесс отдаст команду получить какие то данные и продолжит работать дальше. Асинхронность достигается процедурно за счет коллбеков, промисов (ES6) и async/await (ES8).

*For more info: https://habr.com/ru/company/ruvds/blog/424555/*

1. **Поясните понятие стандартные «системные потоки».**

Методы в глобальном объекте process: stdin(), stdout(), stderr(). Это по порядку стандартный системный поток для ввода, стандартный системный поток для вывода и стандартный системный поток для вывода ошибок.

1. **Поясните назначение функций process.nextTick,** **setImmediate, поясните в чем разница.**

Это величайшая наебка современности, потому что методы похожи, но названия должны поменяться местами.

setImmediate() выполняется на следующем «тике» Event Lopp’a, на следующей его итерации, следующей фазе (если что в ивент лупе тоже есть фазы – обычно выделяют 6), а nextTick() – на той же фазе.

setImmediate() ставит функцию в очередь в за всеми коллбэками I/O операций в очереди событий.

nextTick() ставит функцию в начало очереди событий, чтобы она выполнялась сразу после завершения текущей функции.

setImmediate() выполняется на фазе Check в Ивент Лупе

nextTick() не относится к какой то фазе, существует в отдельном микротаске. Эти микротаски имеют максимальный приоритет и будут выполняться, пока очередь микротасков не кончится. Только после этого будут выполняться макротаски (остальные фазы Ивент Лупа). То есть если запустить рекурсивно nextTick() и написать где нибудь таймер на 5сек, то таймер никогда не выполнится, ибо очередь забита микротаском nextTick()

For more info:

<https://habr.com/ru/company/lineate/blog/585050/>

https://medium.com/devschacht/event-loop-timers-and-nexttick-18579cd122e0