**5. Fundamentals.**

Axios simple example:

const axios = require('axios')

axios.get('https://www.google.com').then((res) => {

    console.log(res);

}).catch((err) => {

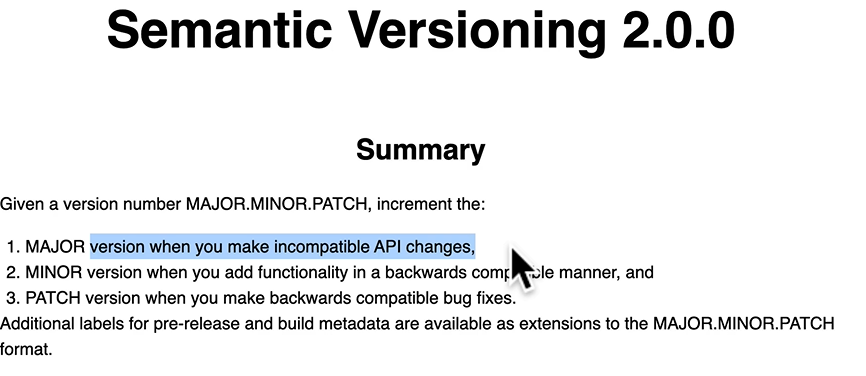
    console.log(err);

}).then(() => {

    console.log('All done');

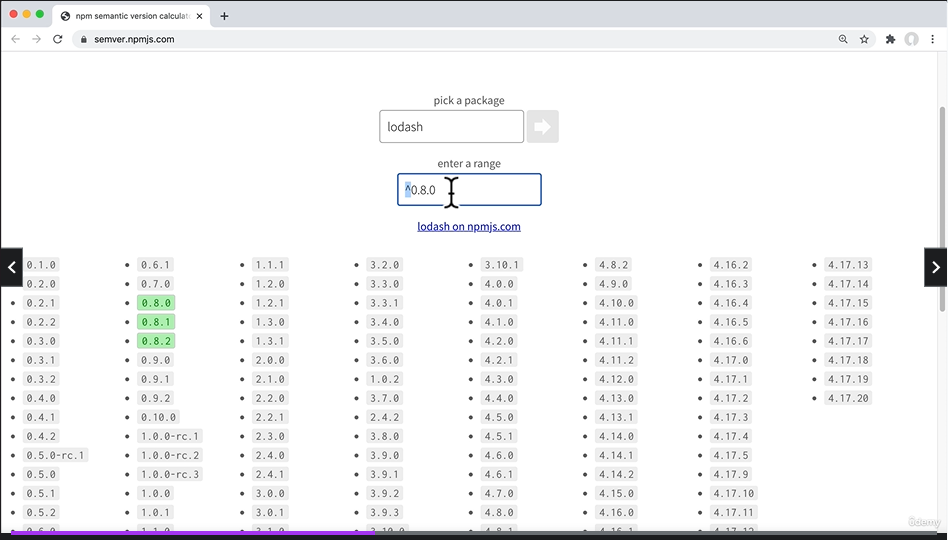
})

**Версионирование** делается по следующему принципу:



*Версия указывается в файле* ***package.json, но только примерная/совместимая. Точная версия указывается в package-lock.json, если есть версия, указанная в этом файле, то ставится именно она. Если только в package.json, то ставится совместимая.***

**Проверить совместимые версии можно на сайте:**



**54. Проблемы секьюрности пакетов NPM**

Обычно при выявлении проблем с секьюрностью, кидает ошибки в терминал, но можно запустить команду **npm audit**, для автоматического фикса: **npm audit fix (меняет версии пакетов!!!)**

**PLANETS PROJECT**

.CSV – Comma separated values

Установим парсер файлов csv: **csv-parse**

**Streaming large database files**

Когда мы хотим прочитать большой файл, загрузить его сразу может быть тяжелой или неподъемной задачей для сервера, поэтому надо использовать СТРИМЫ.

В модуле fs есть метод createReadStream. Он в свою очередь генерирует события: data, end, error

Как соединить парсинг из либы и чтение файла? Через метод **pipe()**, т.к. парсер был разработан с учетом работы в стримах.

const { parse } = require('csv-parse');

const fs = require('fs')

const habitablePlanets = []

function isHabitablePlanets(planet) {

    return planet['koi\_disposition'] === 'CONFIRMED'

        && planet['koi\_insol'] > 0.36 && planet['koi\_insol'] < 1.11

        && planet['koi\_prad'] < 1.6

}

fs.createReadStream('kepler\_data.csv')

    .pipe(parse({  // подключение парсера к стриму

        comment: '#',

        columns: true,

    }))

    .on('data', (planet) => { // обработчик события 'data'

        if (isHabitablePlanets(planet)) { habitablePlanets.push(planet) }

    })

    .on('error', (error) => {

        console.log(error); // обработчик события 'error'

    })

    .on('end', () => { // обработчик события 'end'

        console.log(habitablePlanets.map((planet)=>{

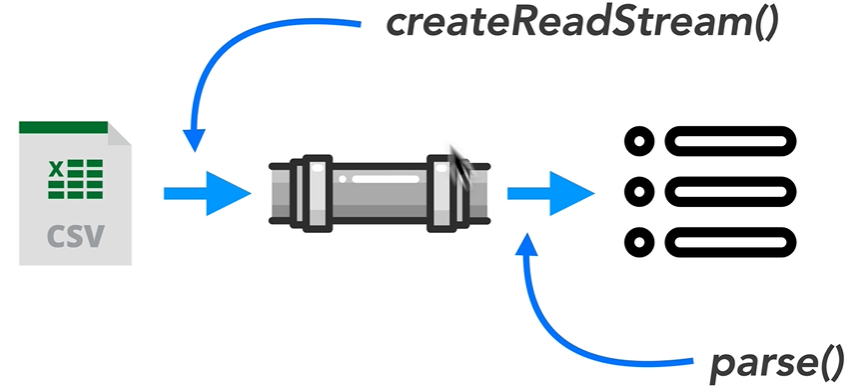
            return planet['kepler\_name']

        }));

        console.log(`=${habitablePlanets.length} habitable planets found!`);

        console.log('done');

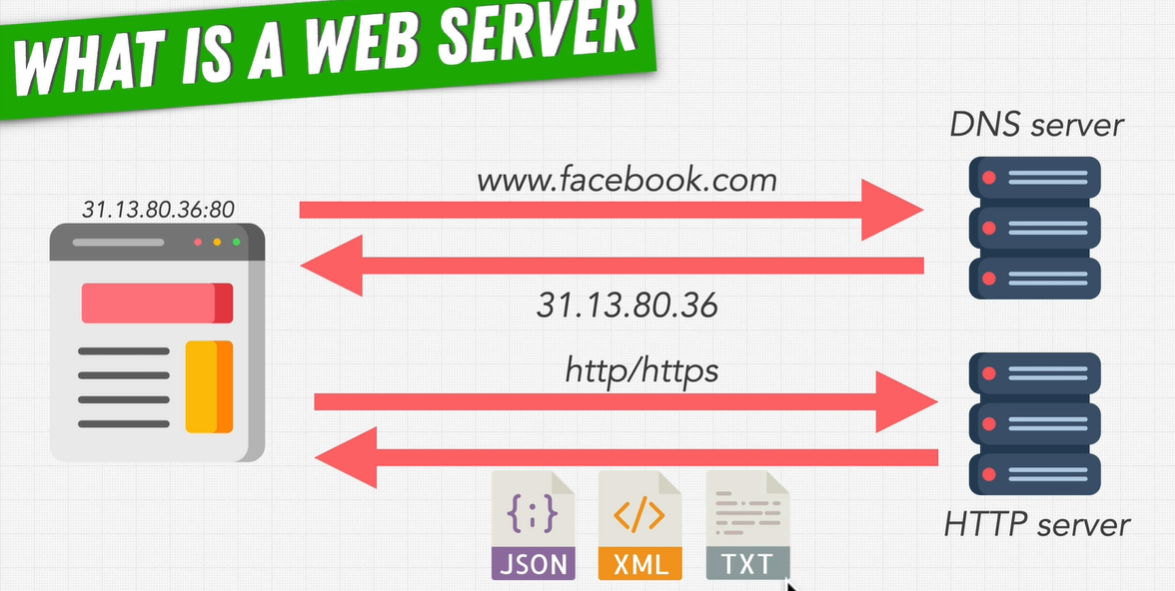
    })



WEB SERVERS

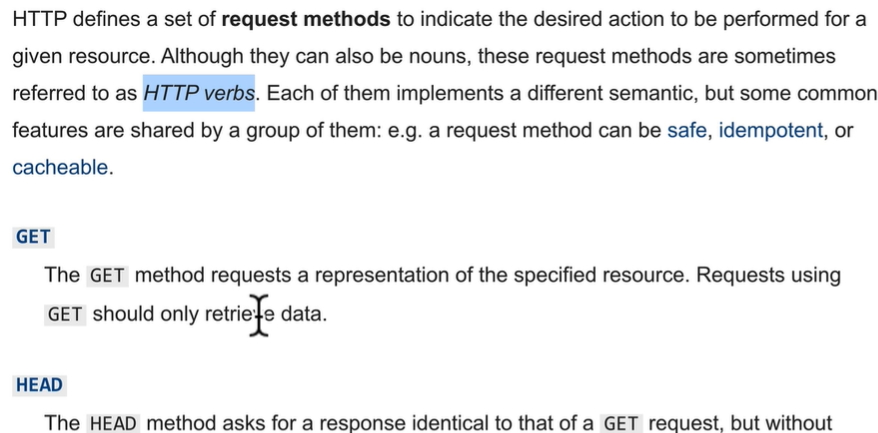
**68. Поток данных**

Схема работы сети:

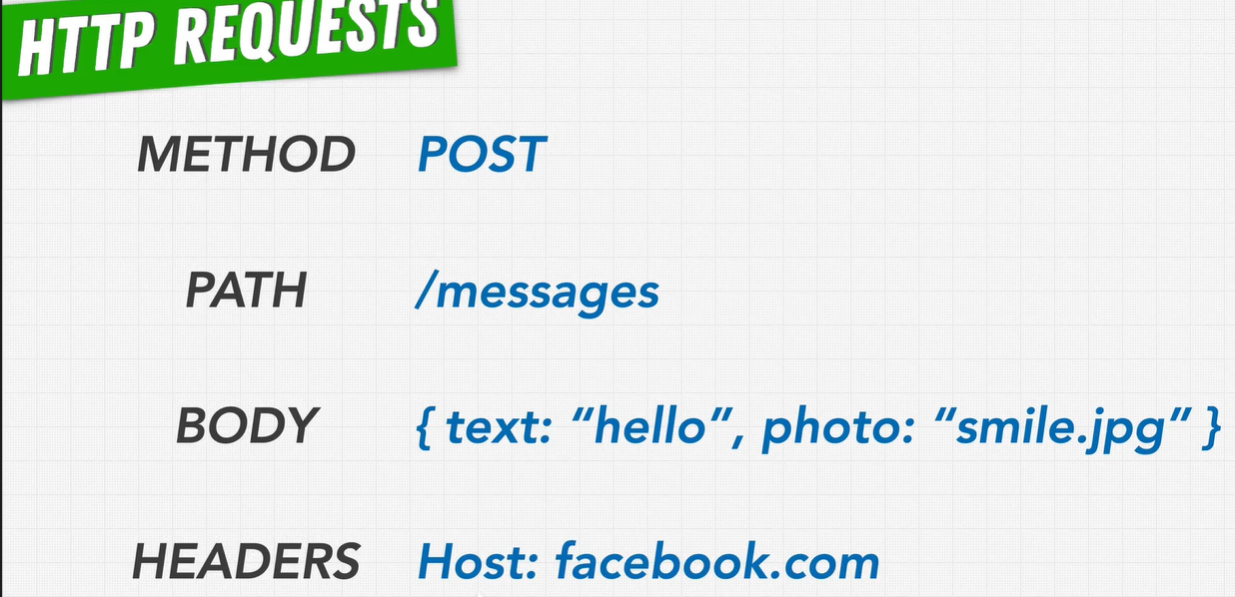


**69. Запросы и ответы**

HTTP сервер может принимать и отвечать только HTTP язык, но бэк может быть написан на любом языке!



**Запрос на сервер состоит из:**



Host – обязательно

**Ответ с сервера состоит из:**

