16. Fetch API va JSON

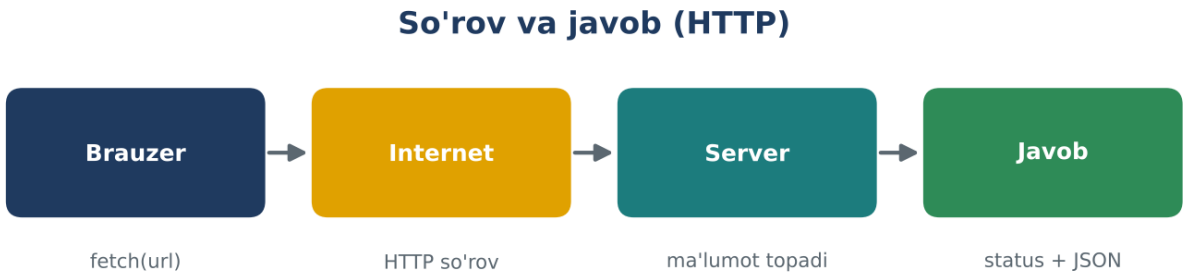


Zamonaviy saytlar ma'lumotni **serverdan** olib turadi: ob-havo, kurslar, postlar. Bu dars ma'lumot formati (**JSON**) va uni olish vositasi (**fetch**) bilan noldan tanishtiradi.

> **Bu darsda nimani o'rganasiz**
>
> Server, so'rov va javob nima.
>
> JSON formati va qoidalari.
>
> `JSON.parse` va `JSON.stringify`.
>
> `fetch` bilan `GET` so'rovi.
>
> Javobni o'qish: `response.json()`.
>
> Ma'lumotni sahifada ko'rsatish.

1.1. Server va so'rov

Veb-sayt ikki tomondan iborat: **mijoz** (brauzer — siz) va **server** (ma'lumot saqlovchi kompyuter). Brauzer **so'rov** (request) yuboradi, server **javob** (response) qaytaradi. Restoran misoli: siz — mijoz, ofitsiant — internet, oshxona — server, buyurtma — so'rov, ovqat — javob.

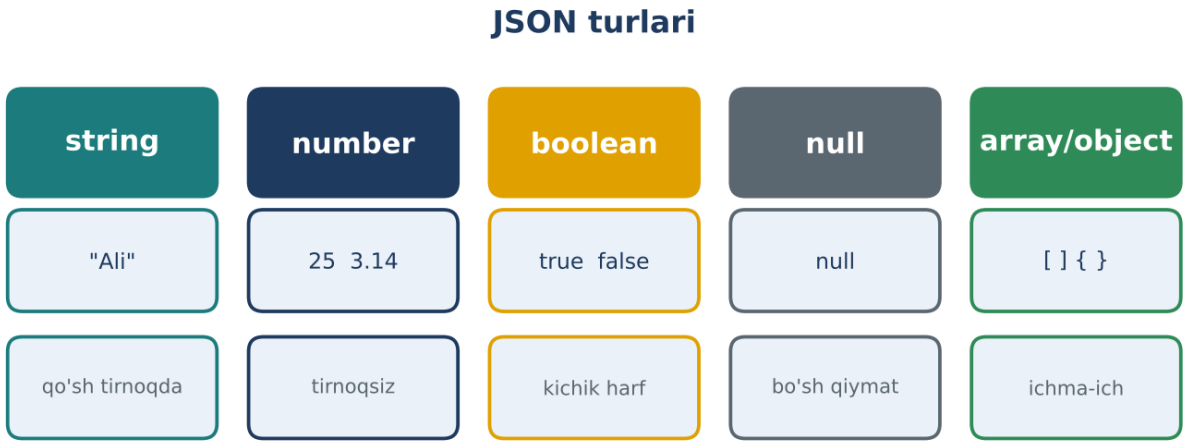


1-rasm. So'rov va javob

So'rov manzili **URL** deb ataladi, masalan `https://api.misol.uz/users`. Bunday ma'lumot beruvchi serverlar **API** (Application Programming Interface) deyiladi. Ma'lumot odatda **JSON** formatida keladi.

1.2. JSON nima?

JSON (JavaScript Object Notation) — ma'lumotni **matn** ko'rinishida saqlash va uzatish formati. U JavaScript obyektiga o'xshaydi, lekin **qat'iyroq** qoidalarga ega.



2-rasm. JSON qiymat turlari

1-jadval. JSON va JS obyekti farqi

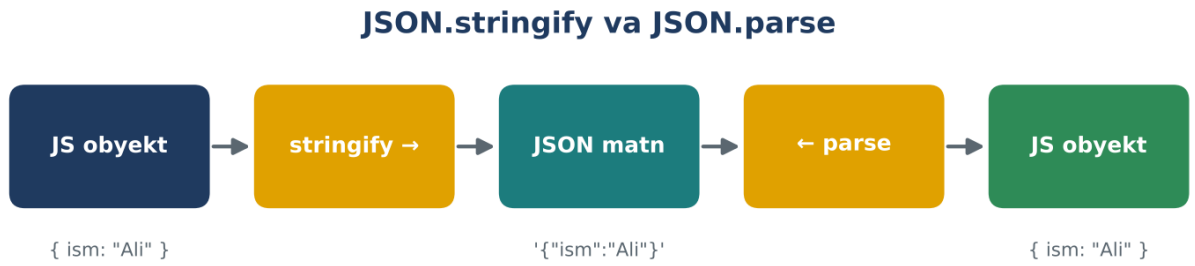
Qoida	JavaScript obyekti	JSON
Kalitlar	`{ ism: "Ali" }` (tirnoqsiz mumkin)	`{ "ism": "Ali" }` (**qo'sh tirnoq shart**)
Satr tirnog'i	`'`, `"` yoki `` ` ``	Faqat `"`
Oxirgi vergul	Mumkin	**Mumkin emas**
Izoh	`//` mumkin	Mumkin emas
Funksiya, `undefined`	Mumkin	Mumkin emas

misol.json

```
{
  "ism": "Ali",
  "yosh": 20,
  "talaba": true,
  "manzil": null,
  "fanlar": ["Matematika", "Informatika"],
  "aloqa": { "telefon": "+998901234567" }
}
```

1.3. JSON.parse va JSON.stringify

Serverdan **matn** keladi. JavaScript bilan ishlash uchun uni **obyektga** aylantirish kerak (`parse`), serverga yuborishda esa **matnga** (`stringify`).



3-rasm. parse va stringify

```
const matn = '{"ism":"Ali","yosh":20,"fanlar":["JS","HTML"]}';

const obyekt = JSON.parse(matn);          // matn ? obyekt
console.log(obyekt.ism, obyekt.fanlar[0]);
console.log(typeof matn, typeof obyekt);

const talaba = { ism: "Vali", yosh: 22, faol: true };
const yangiMatn = JSON.stringify(talaba); // obyekt ? matn
console.log(yangiMatn);
console.log(typeof yangiMatn);

// Chiroyli ko'rinish: 3-parametr — bo'shliqlar soni
console.log(JSON.stringify(talaba, null, 2));
```

Natija:

```
Ali JS
string object
{"ism":"Vali","yosh":22,"faol":true}
string
{
  "ism": "Vali",
  "yosh": 22,
  "faol": true
}
```

Noto'g'ri JSON — xato beradi

```
try {
  JSON.parse("{ism: 'Ali'}");              // kalit tirnoqsiz, bitta tirnoq
} catch (e) {
  console.log(e.name);                     // SyntaxError
}

// stringify da yo'qoladiganlar
console.log(JSON.stringify({ a: 1, f() {}, u: undefined, n: null }));
console.log(JSON.stringify([undefined, () => 1]));
```

Natija:

```
SyntaxError
{"a":1,"n":null}
[null,null]
```

> **Muhim**
>
> `JSON.parse` noto'g'ri matnda **xato tashlaydi**. Server ma'lumotini ishlatayotganda uni `try/catch` ga o'rash yaxshi odat.

1.4. fetch bilan birinchi so'rov

`fetch(url)` — serverga so'rov yuboradi va **Promise** qaytaradi (oldingi mavzuni eslang). Ikki bosqich bor: avval javob sarlavhasi keladi, keyin `response.json()` bilan tana o'qiladi.



4-rasm. fetch bosqichlari

Quyidagi mashqlarda haqiqiy internetga chiqmaymiz: kompyuteringizning o'zida **kichik soxta server** ishga tushiramiz. Brauzerda esa shunchaki haqiqiy API manzilini yozasiz — kod bir xil.

birinchi_fetch.js

```
const http = require("http");
const baza = [{ id: 1, ism: "Ali" }, { id: 2, ism: "Vali" }];
function serverOch() {                        // kichik soxta server (faqat mas
  return new Promise((ok) => {
    const s = http.createServer((req, res) => {
      let tana = "";
      req.on("data", (b) => (tana += b));
```

```
      req.on("end", () => {
        const json = (kod, d) => { res.writeHead(kod, { "Content-Type": "appli
        const id = req.url.split("/")[2];
        if (req.url === "/users" && req.method === "GET") json(200, baza);
        else if (req.url.startsWith("/users/") && req.method === "GET") { cons
        else if (req.url === "/users" && req.method === "POST") { const y = {
        else if (req.url === "/sekin") setTimeout(() => json(200, { ok: 1 }),
        else if (req.url === "/buzuq") { res.writeHead(200); res.end("{buzuq j
        else json(404, { xato: "yo'q" });
      });
    }).listen(0, () => ok({ s, url: "http://localhost:" + s.address().port }))
  });
}

(async () => {
  const { s, url } = await serverOch();

  const javob = await fetch(url + "/users");   // 1-bosqich
  console.log("status:", javob.status, "ok:", javob.ok);
  const data = await javob.json();              // 2-bosqich
  console.log(data);
  console.log("Birinchi:", data[0].ism);

  s.close();
})();
```

Natija:

```
status: 200 ok: true
[ { id: 1, ism: 'Ali' }, { id: 2, ism: 'Vali' } ]
Birinchi: Ali
```

Qisqa yozuv

`const data = await (await fetch(url)).json();` — bir qatorda. Lekin boshlovchi uchun ikki qatorga ajratib yozish tushunarliroq.

then bilan ham yozish mumkin

```
const http = require("http");
const baza = [{ id: 1, ism: "Ali" }, { id: 2, ism: "Vali" }];
function serverOch() {                          // kichik soxta server (faqat mas
  return new Promise((ok) => {
    const s = http.createServer((req, res) => {
      let tana = "";
      req.on("data", (b) => (tana += b));
      req.on("end", () => {
        const json = (kod, d) => { res.writeHead(kod, { "Content-Type": "appli
        const id = req.url.split("/")[2];
        if (req.url === "/users" && req.method === "GET") json(200, baza);
        else if (req.url.startsWith("/users/") && req.method === "GET") { cons
        else if (req.url === "/users" && req.method === "POST") { const y = {
        else if (req.url === "/sekin") setTimeout(() => json(200, { ok: 1 }),
        else if (req.url === "/buzuq") { res.writeHead(200); res.end("{buzuq j
        else json(404, { xato: "yo'q" });
      });
    }).listen(0, () => ok({ s, url: "http://localhost:" + s.address().port }))
```

```
  });
}

serverOch().then(({ s, url }) =>
  fetch(url + "/users/1")
    .then((r) => r.json())
    .then((u) => console.log("Foydalanuvchi:", u.ism))
    .catch((e) => console.log("Xato:", e.message))
    .finally(() => s.close())
);
```

Natija:

```
Foydalanuvchi: Ali
```

1.5. Ma'lumotni sahifada ko'rsatish

Brauzerda olingan ma'lumot odatda ro'yxat sifatida DOM ga chiqariladi (12-mavzudagi usullar):

brauzer.js

```
async function yukla() {
  const javob = await fetch("https://jsonplaceholder.typicode.com/users");
  const users = await javob.json();

  const ul = document.querySelector("#royxat");
  users.forEach((u) => {
    const li = document.createElement("li");
    li.textContent = u.name;              // textContent — xavfsiz
    ul.append(li);
  });
}
yukla();
```

Quyidagi demo jsdom orqali shu ishni **o'zimizdagi** serverdan olingan ma'lumot bilan bajaradi:

dom_fetch.js

```
const http = require("http");
const baza = [{ id: 1, ism: "Ali" }, { id: 2, ism: "Vali" }];
function serverOch() {                        // kichik soxta server (faqat mas
  return new Promise((ok) => {
    const s = http.createServer((req, res) => {
      let tana = "";
      req.on("data", (b) => (tana += b));
      req.on("end", () => {
        const json = (kod, d) => { res.writeHead(kod, { "Content-Type": "appli
        const id = req.url.split("/")[2];
        if (req.url === "/users" && req.method === "GET") json(200, baza);
        else if (req.url.startsWith("/users/") && req.method === "GET") { cons
        else if (req.url === "/users" && req.method === "POST") { const y = {
        else if (req.url === "/sekin") setTimeout(() => json(200, { ok: 1 }),
        else if (req.url === "/buzuq") { res.writeHead(200); res.end("{buzuq j
        else json(404, { xato: "yo'q" });
```

```
      });
    }).listen(0, () => ok({ s, url: "http://localhost:" + s.address().port }));
  });
}

(async () => {
  const { s, url } = await serverOch();
  const users = await (await fetch(url + "/users")).json();

  const ul = document.querySelector("#royxat");
  users.forEach((u) => {
    const li = document.createElement("li");
    li.textContent = u.id + ". " + u.ism;
    ul.append(li);
  });
  console.log(document.body.innerHTML);
  s.close();
})();
```

Natija:

```
<ul id="royxat"><li>1. Ali</li><li>2. Vali</li></ul>
```

1.6. Eng ko'p uchraydigan xatolar

2-jadval. Tipik xatolar

Xato	Sababi	To'g'ri yozuv
`response.json()` ni `await` siz	Promise obyekti keladi	`await javob.json()`
`fetch` dan keyin to'g'ridan-to'g'ri `.ism`	Javob obyekti, ma'lumot emas	Avval `.json()`
JSON da bitta tirnoq	`SyntaxError`	Faqat "
Obyektni matn deb ishlatish	`[object Object]`	`JSON.stringify(obj)`
`JSON.parse(obyekt)`	Matn kutadi	Matnni bering

```
const o = { a: 1 };
console.log("Salom " + o);                  // [object Object]
console.log("Salom " + JSON.stringify(o));  // to'g'ri
```

Natija:

```
Salom [object Object]
Salom {"a":1}
```

1.7. Mashqlar

1. O'zingiz haqingizda JSON yozing (ism, yosh, fanlar massivi, manzil obyekti).
2. Shu JSON ni `JSON.parse` bilan o'qib, ismni chiqaring.
3. Obyektni `JSON.stringify(obj, null, 2)` bilan chiroyli chiqaring.

4. Noto'g'ri JSON bilan `try/catch` ni sinab ko'ring.
5. `fetch` bilan ro'yxat oling va `console.log` qiling.
6. Olingan ro'yxatni `<ul>` ga chiqaring.
7. Faqat ismlarni massiv qilib (`map`) chiqaring.
8. Yosh bo'yicha filtrlang (`filter`).

1.8. O'zingizni tekshiring

3-jadval. Qisqa test

№	Savol	Javob
1	JSON nima?	Ma'lumotni matn sifatida saqlash formati
2	JSON da kalit tirnog'i?	Faqat qo'sh tirnoq
3	Matnni obyektga?	`JSON.parse`
4	Obyektni matnga?	`JSON.stringify`
5	`fetch` nima qaytaradi?	Promise
6	Javob tanasini JSON qilib o'qish?	`response.json()`
7	`response.json()` ham Promisemi?	Ha, `await` kerak
8	API nima?	Dastur bilan muloqot interfeysi, ma'lumot beruvchi manzil

6 va undan ko'p to'g'ri javob — **2-darajaga** tayyorsiz.

> **Xulosa**
>
> JSON — ma'lumot matni; `parse` uni obyektga, `stringify` obyektni matnga aylantiradi. `fetch` Promise qaytaradi; javobni `await response.json()` bilan o'qiymiz.

> **Chuqurroq o'rganmoqchimisiz?**
>
> Bu mavzuning o'rta va murakkab darajalari JavaScript adaptiv kursida — daraja testi sizga mos materialni o'zi tanlaydi, oxirida sertifikat.