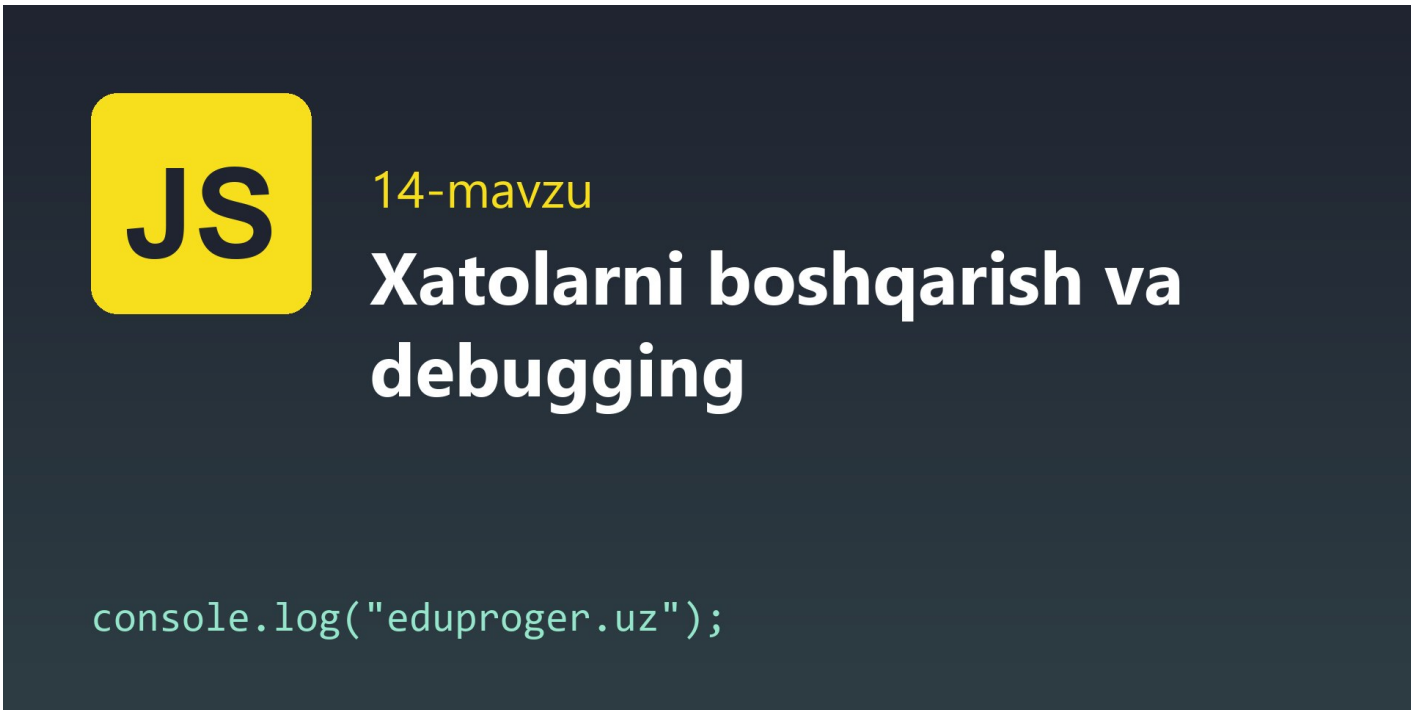
14. Xatolarni boshqarish va debugging



Xato — dasturlashning **normal qismi**. Professional dasturchi ham har kuni xato qiladi; farq shundaki, u xatoni **tez topadi va to'g'ri qayta ishlaydi**. Bu dars xatolarning tilini o'rgatadi.

> **Bu darsda nimani o'rganasiz**
>
> Xato nima va u turlari (sintaksis, ish vaqti, mantiqiy).
>
> Xato xabarini o'qish.
>
> `console.log`, `console.table` va boshqa vositalar.
>
> `try...catch...finally`.
>
> `throw` bilan o'zimiz xato yaratish.
>
> Eng oddiy debugging usullari.

1.1. Xato nima?

Dastur ba'zan **kutilganidek ishlamaydi**. Sabab har xil bo'lishi mumkin: harf xatosi, mavjud bo'lmagan narsani ishlatish, noto'g'ri mantiq. Xatolar uch guruhga bo'linadi:

1-jadval. Xato guruhlari

Guruh	Qachon bilinadi	Misol
Sintaksis xatosi	Kod ishga tushishidan **oldin**	Qavs yopilmagan, `consol.log(` kabi yozuv xatolari
Ish vaqti xatosi (runtime)	Dastur **ishlayotganda**	Mavjud bo'lmagan o'zgaruvchini ishlatish, `null.x`
Mantiqiy xato	Hech qachon "qizil" xato bermaydi — natija **noto'g'ri**	Yoshni hisoblashda + o'rniga - yozish

> **Eng xavfli turi**
>
> **Mantiqiy xato** eng qiyini: dastur ishlaydi, hech narsa buzilmaydi, lekin javob **noto'g'ri**. Shuning uchun natijani doim tekshirish kerak.

1.2. Xato xabarini o'qish

Brauzer va Node.js xatoni **tushuntirib** beradi. Xabar odatda uch qismdan iborat: **xato turi**, **izoh**, **qayerda**. Uni o'qishni o'rganish — eng foydali ko'nikma.

```
try {
  console.log(ism);                 // e'lon qilinmagan
} catch (e) {
  console.log(e.name);              // xato turi
  console.log(e.message);           // izoh
}

try {
  const odam = null;
  console.log(odam.yosh);
} catch (e) {
  console.log(e.name + ": " + e.message);
}

try {
  const son = 5;
  son();                            // son — funksiya emas
} catch (e) {
  console.log(e.name + ": " + e.message);
}

try {
  const c = 1;
  c = 2;
} catch (e) {
  console.log(e.name + ": " + e.message);
}
```

Natija:

```
ReferenceError
ism is not defined
TypeError: Cannot read properties of null (reading 'yosh')
```

```
TypeError: son is not a function
TypeError: Assignment to constant variable.
```

2-jadval. Eng ko'p uchraydigan xabarlar

Xabar	Nimani anglatadi	Odatiy sabab
`ReferenceError: x is not defined`	x nomli o'zgaruvchi yo'q	Harf xatosi, e'lon qilinmagan, scope tashqarisi
`TypeError: Cannot read properties of undefined (reading 'x')`	undefined ning x xossasini o'qimoqchisiz	Obyekt kutilgan edi, lekin `undefined/null` keldi
`TypeError: x is not a function`	x funksiya emas, lekin chaqirilgan	Nom xatosi yoki noto'g'ri tur
`TypeError: Assignment to constant variable`	const ni qayta tayinlash	`let` kerak
`SyntaxError: Unexpected token ...`	Yozuv noto'g'ri	Qavs, vergul, tirnoq xatosi
`RangeError: Maximum call stack size exceeded`	Funksiya cheksiz o'zini chaqirgan	Rekursiyada to'xtash sharti yo'q

> **Xabarni o'qing!**
>
> Boshlovchilar xatoni ko'rib vahimaga tushadi va o'qimaydi. Aslida xabarda javobning yarmi bor: **birinchi qatorni** o'qing va **qator raqamiga** qarang.

1.3. console vositalari

`console.log` — eng oddiy "tekshirish" asbobi: o'zgaruvchi qiymatini ko'rsatadi. Boshqa metodlar ham bor:

3-jadval. console metodlari

Metod	Vazifasi
`console.log(...)`	Oddiy chop etish
`console.error(...)`	Xato sifatida (qizil, brauzerda)
`console.warn(...)`	Ogohlantirish (sariq)
`console.table(massiv)`	Jadval ko'rinishida
`console.group / groupEnd`	Guruhlab, chekinish bilan
`console.time / timeEnd`	Vaqtni o'lchash
`console.count(nom)`	Necha marta chaqirilganini sanash

Metod	Vazifasi
`console.assert(shart, xabar)`	Shart noto'g'ri bo'lsa xabar beradi

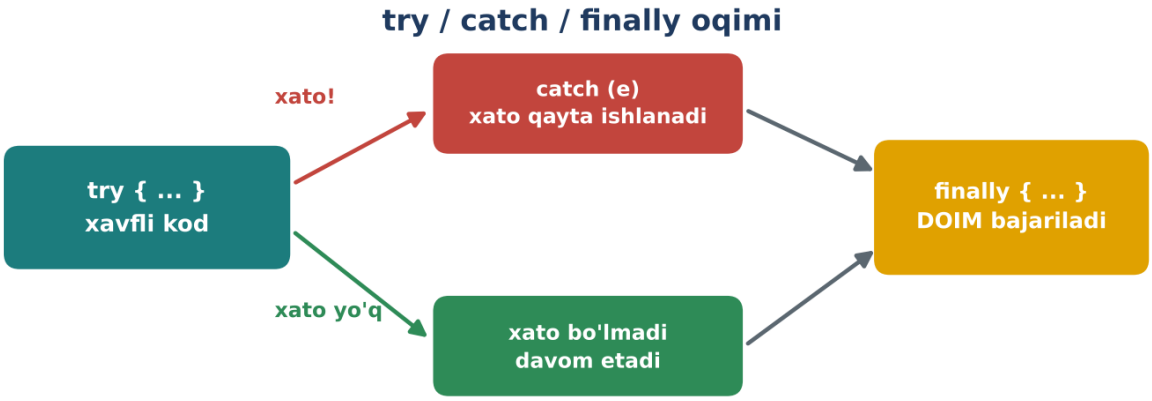
```
const talabalar = [{ ism: "Ali", ball: 85 }, { ism: "Vali", ball: 60 }];

console.log("Talabalar:", talabalar);
console.table(talabalar);

console.group("Hisobot");
console.log("Jami:", talabalar.length);
console.log("O'rtacha:", (85 + 60) / 2);
console.groupEnd();

console.count("chaqiruv");
console.count("chaqiruv");

// Bir nechta qiymatni belgilab ko'rish: obyekt ichiga qo'ying
const x = 5, y = 10;
console.log({ x, y });
```

Natija:

```
Talabalar: [ { ism: 'Ali', ball: 85 }, { ism: 'Vali', ball: 60 } ]
???????????????????????????
? (index) ? ism    ? ball ?
???????????????????????????
? 0       ? 'Ali'  ? 85   ?
? 1       ? 'Vali' ? 60   ?
???????????????????????????
Hisobot
  Jami: 2
  O'rtacha: 72.5
chaqiruv: 1
chaqiruv: 2
{ x: 5, y: 10 }
```

1.4. try ... catch ... finally

Xato bo'lsa dastur **to'xtab qoladi**. Lekin ba'zan xatoni **kutamiz** (masalan, foydalanuvchi noto'g'ri ma'lumot kiritishi mumkin) va dastur davom etishini xohlaymiz. Buning uchun `try...catch`:



1-rasm. try / catch / finally oqimi

```
try {
  console.log("1. try boshlandi");
  const x = JSON.parse("{ buzuq json");
  console.log("2. bu qator ishlamaydi");
} catch (xato) {
  console.log("3. catch:", xato.name);
} finally {
  console.log("4. finally — doim ishlaydi");
}
console.log("5. dastur davom etdi");
```

Natija:

```
1. try boshlandi
3. catch: SyntaxError
4. finally — doim ishlaydi
5. dastur davom etdi
```

- **try** — xavfli kodni shu yerga yozasiz.
- **catch (e)** — xato bo'lsa **faqat** shunda ishlaydi; e — xato obyekti.
- **finally** — xato bo'ldimi, yo'qmi, **doim** bajariladi (tozalash uchun).

catch ni bo'sh qoldirmang

`catch (e) { }` — xatoni **yutib yuboradi**; keyin nima uchun buzilganini topa olmaysiz. Hech bo'lmasa xatoni loglang.

1.5. throw: o'zimiz xato yaratamiz

Noto'g'ri holat yuzaga kelganini bilsangiz, `throw new Error("xabar")` bilan **o'zingiz** xato chaqirishingiz mumkin:

```
function bol(a, b) {
  if (b === 0) {
    throw new Error("Nolga bo'lib bo'lmaydi!");
```

```
  }
  return a / b;
}

try {
  console.log(bol(10, 2));
  console.log(bol(5, 0));
} catch (e) {
  console.log("Xato ushlandi:", e.message);
}

function yoshTekshir(yosh) {
  if (typeof yosh !== "number") throw new TypeError("Yosh son bo'lishi kerak")
  if (yosh < 0 || yosh > 150) throw new RangeError("Yosh 0–150 oralig'ida bo'l
  return true;
}
for (const q of [25, "yigirma", 200]) {
  try { yoshTekshir(q); console.log(q, "— to'g'ri"); }
  catch (e) { console.log(e.name + ":", e.message); }
}
```

Natija:

```
5
Xato ushlandi: Nolga bo'lib bo'lmaydi!
25 — to'g'ri
TypeError: Yosh son bo'lishi kerak
RangeError: Yosh 0–150 oralig'ida bo'lishi kerak
```

1.6. Oddiy debugging usullari

1. **Xabarni o'qing.** Qaysi xato turi, qaysi qator?
2. **`console.log` bilan tekshiring.** Shubhali o'zgaruvchi qiymatini chop eting: u siz o'ylagandek emasmi?
3. **Qadamma-qadam qisqartiring.** Kodning yarmini o'chirib ko'ring: xato qayerda yo'qoladi?
4. **Xatoni takrorlang.** Qanday qilsangiz xato chiqishini aniq biling.
5. **Ertasi kuni qarang.** Ba'zan dam olib, yangi nazar bilan qarash yordam beradi.

```
// Xato: o'rtacha noto'g'ri chiqyapti. Sababi nima?
function ortacha(arr) {
  let j = 0;
  for (let i = 0; i <= arr.length; i++) {         // <= — xato!
    j += arr[i];
  }
  return j / arr.length;
}
console.log(ortacha([2, 4, 6]));                  // NaN

// console.log bilan tekshiruv
function tekshir(arr) {
  let j = 0;
  for (let i = 0; i <= arr.length; i++) {
    console.log("i =", i, "arr[i] =", arr[i], "j =", j);
    j += arr[i];
```

```
  }
}
tekshir([2, 4, 6]);
// Sabab topildi: i = 3 da arr[3] = undefined, j NaN bo'lib ketdi. Tuzatish: i
```

Natija:

```
NaN
i = 0 arr[i] = 2 j = 0
i = 1 arr[i] = 4 j = 2
i = 2 arr[i] = 6 j = 6
i = 3 arr[i] = undefined j = 12
```

1.7. Kod izi

```
try {
  console.log("A");
  throw new Error("x");
  console.log("B");
} catch (e) {
  console.log("C");
} finally {
  console.log("D");
}
console.log("E");
```

4-jadval. Kod izi

Qadam	Nima bo'ladi	Chiqadi
1	`try` boshlandi	A
2	`throw` — xato, `try` darrov uziladi	—
3	`B` bajarilmaydi, `catch` ga o'tildi	C
4	`finally` doim	D
5	Dastur davom etdi	E

1.8. Mini loyihalar

xato_loyiha.js

```
// 1) Xavfsiz JSON o'qish
function xavfsizParse(matn, standart = null) {
  try { return JSON.parse(matn); }
  catch { return standart; }
}
console.log(xavfsizParse('{"a":1}'), xavfsizParse("buzuq"), xavfsizParse("buzu

// 2) Xavfsiz bo'lish
function xavfsizBol(a, b) {
  try {
    if (b === 0) throw new Error("Nolga bo'lish");
    return a / b;
  } catch (e) {
```

```
    return `Xato: ${e.message}`;
  }
}
console.log(xavfsizBol(10, 4), xavfsizBol(1, 0));

// 3) Son o'qish va tekshirish
function sonOqi(matn) {
  const n = Number(matn);
  if (matn.trim() === "" || Number.isNaN(n)) throw new Error(`"${matn}" son em
  return n;
}
for (const q of ["42", "3.5", "abc", ""]) {
  try { console.log(sonOqi(q)); } catch (e) { console.log(e.message); }
}

// 4) finally bilan tozalash
let resurs = "ochiq";
try { throw new Error("buzildi"); }
catch (e) { console.log("Xato:", e.message); }
finally { resurs = "yopiq"; }
console.log("resurs:", resurs);
```

Natija:

```
{ a: 1 } null {}
2.5 Xato: Nolga bo'lish
42
3.5
"abc" son emas
"" son emas
Xato: buzildi
resurs: yopiq
```

1.9. Eng ko'p uchraydigan xatolar (va ular haqida)

5-jadval. Tipik xatolar

Xato	Sababi	Tuzatish
`consol.log` / `Console.log`	Harf xatosi: JS katta-kichik harfni farqlaydi	`console.log`
`catch` ni bo'sh qoldirish	Xato yashirinib qoladi	Hech bo'lmasa `console.log(e)`
Hamma narsani bitta katta `try` ga o'rash	Xato qayerdaligi noaniq	Faqat xavfli qismni o'rang
`throw "matn"` yozish	Xato obyekti emas, `stack` yo'q	`throw new Error("matn")`
`try` ichida `let` e'lon qilib, tashqarida ishlatish	Blok scope	O'zgaruvchini `try` dan oldin e'lon qiling
Sintaksis xatosini `try/catch` bilan ushlamoqchi bo'lish	Kod ishga tushmaydi, `catch` ga yetmaydi	Yozuvni to'g'rilang

1.10. Mashqlar

1. Mavjud bo'lmagan o'zgaruvchini chop eting va xabarni o'qib, turini ayting.
2. `null.ism` ni `try/catch` bilan ushlang va xabarni chiqaring.
3. `kvadratIldiz(n)` yozing: manfiy son uchun `throw new Error(...)`.
4. `try/catch/finally` bilan A, B, C chiqadigan dastur yozing: tartibini oldindan ayting.
5. `console.table` bilan 3 ta mahsulotni jadval ko'rinishida chiqaring.
6. `JSON.parse("{ buzuq")` ni xavfsiz o'rovchi funksiya yozing.
7. Massiv elementlari o'rtachasi funksiyasidagi <= xatosini toping.
8. `console.assert(2 + 2 === 5, "Hisob xato")` ni sinab ko'ring.

1.11. O'zingizni tekshiring

6-jadval. Qisqa test

№	Savol	Javob
1	Eng xavfli xato turi?	Mantiqiy (xabar bermaydi)
2	`ReferenceError` nima?	E'lon qilinmagan nom ishlatilgan
3	`try` da xato bo'lsa?	`catch` ishlaydi
4	`finally` qachon ishlaydi?	Doim
5	Xato yaratish?	`throw new Error("...")`
6	Xabarni qayerdan o'qish?	`e.message`
7	`undefined.x` qanday xato?	`TypeError`
8	Jadval ko'rinishida chop etish?	`console.table`

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

> **Xulosa**
>
> Xato — normal hodisa. Xabarni o'qing (tur, izoh, qator), `console.log` bilan tekshiring, kutilgan xatolarni `try/catch` bilan ushlang, noto'g'ri holatda `throw new Error`. `finally` — tozalash uchun.

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