**1. Getting Started**

4. Installing & Using TypeScript

**Typescript**-ის დასაინსტალირებლად ვიყენებთ ბრძანებას:

npm install -g typescript

შემდეგ კი ბრძანებით - **tsc  <ჯს ფაილი>**- დააკომპაილებს **TS**-ს **JS**-ში.

**1. Getting Started**

9. The Course Project Setup

ლაივ სერვერისთვის შეგვიძლია **npm**-ის ინიციალიზების მერე, დავაინსტალიროთ სპეციალური პაკეჯი - npm install --save-dev lite-server

შემდეგ კი უნდა გავუშვათ ბრძანება: **lite-server** და ლაივი გაეშვება;

**2. TypeScript Basics & Basic Types**

16. Object Types

**TS**-ში არის პრიმიტიული ტიპები:

**- number**

**- string**

**- boolean**

და ა.შ.

თუ **const**-ში ვინახავთ დეითას, ამ ცვლად ავტომატურად მიენიჭება არა მხოლოდ დეითას ტიპი, არამედ თავად დეითაც, რამდენადაც ესაინმენტი არ ხდება.

თუ **let**-ში ვინახავთ, ამ ცვლადს მიენიჭება მხოლოდ დეითას ტიპი, ანუ სხვა დეითას ესაინმენტისას არ მიიღებს.

**JS**-სსაც აქვს ტიპები, თუმცა ისინი რანთაიმში მოწმდება, **TS**-ის შემთხვევაში კი - დეველოპმენტ თაიმში.

**2. TypeScript Basics & Basic Types**

22. Union Types

ობიექტის ტიპების დეკლარირება რამდენიმენაირად შეიძლება:

1. const person = {
2. name: 'Maxi',
3. age: 30,
4. hobbies: ['Sports', 'Cooking']
5. }
7. const person: object = {
8. name: 'Maxi',
9. age: 30,
10. hobbies: ['Sports', 'Cooking']
11. }
13. const person: {} = {
14. name: 'Maxi',
15. age: 30,
16. hobbies: ['Sports', 'Cooking']
17. }
19. const person: {
20. name: string;
21. age: number; // ასევე შეგვიძლია უბრალოდ რიცხვი დავწეროთ და მის მეტს არ მიიღებს, მაგ - 30
22. hobbies: string[];
23. } = {
24. name: "Maxi",
25. age: 30,
26. hobbies: ["Sports", "Cooking"],
27. };

**2. TypeScript Basics & Basic Types**

22. Union Types

**Tuples**:

1. // Tuples = ტიპი => ფიქსირებული მასივისთვის მაგალითად => (string | number)[] = [1, 'gio]
2. // .push() დაშვებულია ტიუპელებში
4. const personTuple: {
5. name: string;
6. age: number;
7. hobbies: string[];
8. role: [number, string]; // Tuples
9. } = {
10. name: "Maxi",
11. age: 30,
12. hobbies: ["Sports", "Cooking"],
13. role: [1, "gio"],
14. };

**2. TypeScript Basics & Basic Types**

22. Union Types

**Enums:**

1. // Enum = ტიპი => enum {NEW, OLD}
3. enum Role {
4. ADMIN,
5. READ\_ONLY,
6. AUTHOR,
7. }; // ისინი ავტომატურად დაინომრებიან 0++, თუმცა ხელით მანიპულირებაც შეიძლება
9. const personEnum = {
10. name: "Maxi",
11. age: 30,
12. hobbies: ["Sports", "Cooking"],
13. role: Role.ADMIN,
14. };

**2. TypeScript Basics & Basic Types**

22. Union Types

**Union Types: number | string :**

1. // Union Type
2. function combine(input1: number | string, input2: number | string) {
3. let result;
4. if (typeof input1 === "number" && typeof input2 === "number") {
5. result = input1 + input2;
6. }else {
7. result = input1.toString() + input2.toString()
8. }
9. return result;
10. }