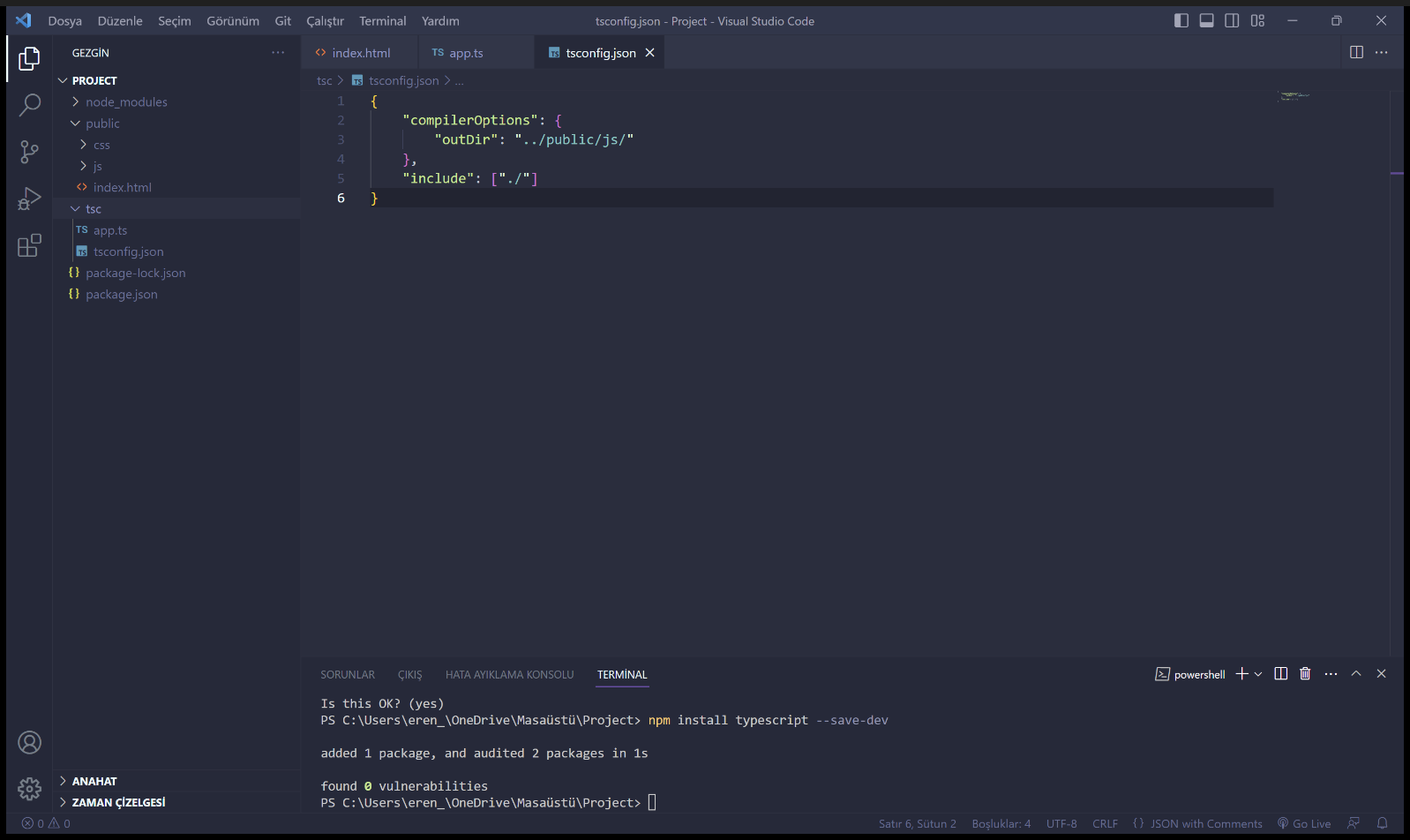
Geliştirme Ortamının Hazırlanması

Terminale aşağıdaki komutu yazıp typescript’i projemize ekliyoruz.

npm install typescript --save-dev

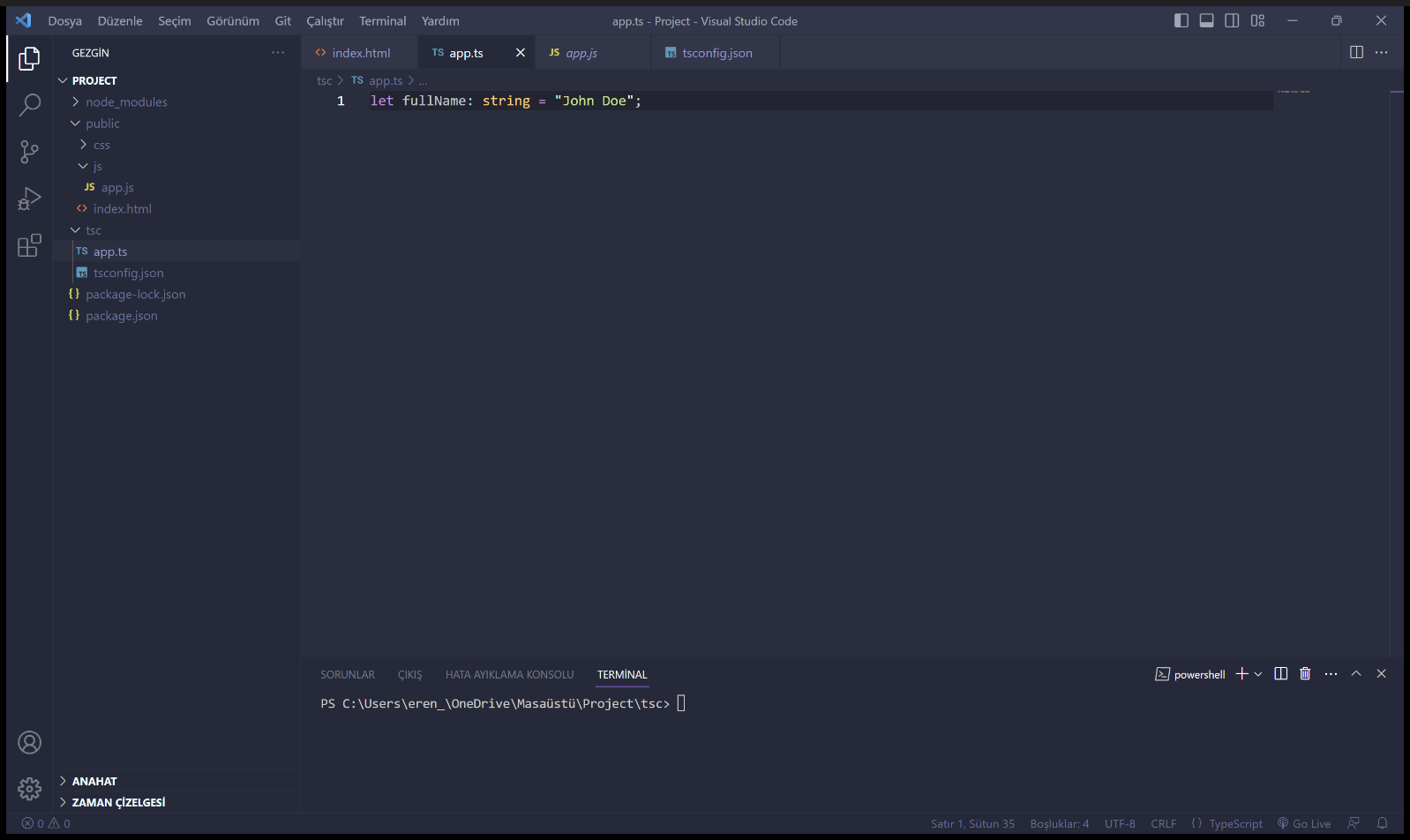
tsc isimli klasörün içerisinde typescript’in konfigürasyon bilgilerinin tutulduğu tsconfig.json dosyasını oluşturuyoruz.



outDir parametresi derlenen dosyaların nereye çıkarılacağını belirlemektedir.

include parametresi derleyiciye typescript dosyalarını nerede bulacağını söylemektedir.

app.ts içerisine typescript kodlarımızı yazıyoruz.



Aşağıdaki komut ile typescript dosyasını derliyoruz.

npx tsc -p tsconfig.json -w

-w(watch) parametresi değişiklik yapılduğında dosyanın otomatik olarak yeniden derlenmesi için kullanılmaktadır.

Variables

Typescript’de değişkenler aşağıdaki gibi tanımlanmaktadır.

Example

let fullName: string = "John Doe";

let age: number = 30;

* Değişken tanımlarken birden fazla tip belirtebilirsiniz.

Example

let data: string | number | boolean;

data = "0";

data = 1;

data = false;

console.log(data) //false

Arrays

Typescript’de diziler iki farklı şekilde tanımlanabilmektedir.

Example

//Yöntem 1

let fruits: string[] = ["Apple", "Cherry", "Melon"];

//Yöntem 2

let colors: Array<string> = ["Red", "Green", "Blue"];

* Önce diziyi tanımlayıp değer atamasını daha sonra yapabilirsiniz.

Example

let numbers: number[];

numbers = [1, 2, 3, 4, 5, 6];

* Dizi tanımlarken birden fazla tip belirtebilirsiniz.

Example

let arr: (string | number)[] = [1, 2, 3, "A", "B", "C"];

let arr: Array<string | number> = [1, 2, 3, "A", "B", "C"];

Methods

Geriye değer döndürmeyen metodlar void olarak tanımlanmaktadır.

Example

function sayHello(): void{

console.log("Hi");

}

Geriye değer döndüren metodlar aşağıdaki gibi tanımlanmaktadır.

Example

function addNumbers(x: number, y: number): number{

return x + y;

}

console.log(addNumbers(2, 4)); //6

* Metodların parametrelerine default değerler verilebilmektedir.

Example

function write(name: string, age:number = null): void{

console.log(name, age);

}

write("John Doe"); //John Doe null

write("John Doe", 30); //John Doe 30

* Parametreler opsiyonel olarak tanımlanabilmektedir.

Example

function writeNumbers(x: number, y: number, z?: number): void{

console.log(x, y, z); //1 2 undefined

}

writeNumbers(1, 2);

* Metodlar aşırı yüklenebilmektedir.(overloading)

Example

//Önce metodların imzaları tanımlanmaktadır.

function add(a: number, b: number): number;

function add(a: string, b: string): string;

//Daha sonra metodun kendisi tanımlanmaktadır.

function add(a: any, b: any): any{

return a + b;

}

console.log(add(1, 2)); //3

console.log(add("EREN", "SATI")); //ERENSATI

(Rest Parameters)

Example

function add(...numbers: number[]): number{

let total = 0;

numbers.forEach((number) => total += number);

return total;

}

console.log(add()); //0

console.log(add(10)); //10

console.log(add(10, 20)); //30

console.log(add(10, 20, 30)); //60

Example

function add(message: string, ...numbers: number[]): number{

console.log(message);

let total = 0;

numbers.forEach((number) => total += number);

return total;

}

Data Type - any

any type her türden veriyi kabul etmektedir.

* Değişkenler üzerinde kullanılabilmektedir.

Example

let something: any;

* Diziler üzerinde kullanılabilmektedir.

Example

let arr: any[];

Data Type - object

Example

type employee = {

fullName: string,

age: number,

salary: number

}

let Employee: employee = {

fullName: "John Doe",

age: 30,

salary: 5000

}

Example

type person = {

fullName: string,

age: number

}

type employee = {

salary: number

}

let obj: person | employee = {

salary: 5000

}

console.log(obj); //Object { salary: 5000 }

Example

type person = {

fullName: string,

age: number

}

type employee = {

salary: number

}

let obj: person & employee = {

fullName: "John Doe",

age: 30,

salary: 4000

}

console.log(obj); //Object { fullName: "John Doe", age: 30, salary: 5000 }

Tuple

Bir tuple aşağıdaki gibi tanımlanmaktadır.

Example

let employee: [number, string] = [1, "John"];

* Aşağıdaki gibi tuple dizisi tanımlayabilirsiniz. Tuple dizileri typescript tarafından çok boyutlu dizilere dönüştürülmektedir.

Example

let employee: [number, string][] = [[1, "John"], [2, "Elise"], [3, "Julia"]];

Class

Bir class aşağıdaki gibi tanımlanmaktadır.

Example

class Class1{

}

* Class içinde field ve metod tanımlayabilirsiniz.

Example

class Person{

fullName: string;

age: number = null; //Özelliklere varsayılan değer verebilirsiniz.

constructor(fullName: string, age: number){

this.fullName = fullName;

this.age = age;

}

showInfos(){

console.log(this.fullName, this.age);

}

}

* Metod ve field’lar erişim belirteçlerine sahip olabilirler. (public, protected, private)
* Bir özelliğin sadece okunabilir olmasını istiyorsanız readonly olarak tanımlayabilirsiniz. Readonly olarak tanımlanan özelliklerin değerleri sadece constructor içerisinden atanabilmektedir.

Example

class Person{

public readonly id: number;

fullName: string;

constructor(id: number, fullName: string){

this.id = id;

this.fullName = fullName;

}

}

* Class’lar birbirlerini miras alabilirler.

Example

class Person{

fullName: string;

age: number;

constructor(fullName: string, age: number){

this.fullName = fullName;

this.age = age;

}

}

class Employee extends Person{

salary: number;

constructor(fullName: string, age: number, salary: number){

super(fullName, age);

this.salary = salary;

}

}

* Class içerisinde static metod ve property tanımlayabilirsiniz.

Example

class Class1{

constructor(){

Class1.Increment();

}

static Count = 0;

static Increment(){

this.Count++;

}

}

let obj1 = new Class1();

let obj2 = new Class1();

console.log(Class1.Count); //2

Enum

Aşağıdaki örnekte number türünde bir enum tanımladık.

Example

enum Seasons{

autumun,

winter,

spring,

summer

}

console.log(Seasons.autumun); //0

console.log(Seasons[1]); //winter

* Elemanların sıra değerlerini = operatörü ile ayarlayabilirsiniz.

Example

enum Seasons{

autumun = 2,

winter = 4,

spring = 6,

summer = 8

}

console.log(Seasons.autumun); //2

console.log(Seasons.winter); //4

console.log(Seasons.spring); //6

console.log(Seasons.summer); //8

Aşağıdaki örnekte string türünde bir enum tanımladık.

Example

enum Media{

Newspaper = "NEWSPAPER",

Newsletter = "NEWSLETTER",

Magazine = "MAGAZINE",

Book = "BOOK"

}

console.log(Media.Newspaper); //NEWSPAPER