\_Cài dặt typescript: npm install -g typescript.

\_ Dịch TS sang JS: tsc tten\_file.ts --watch

\_ Kiểu dữ liệu:

**+String**: **let** fullName: string = `Bob Bobbington`;

**let** sentence: string = `Hello, my name is ${ fullName }.

**+number**: **let** decimal: number = 6;

**+Enum: enum** Color {Red = 1, Green, Blue}

**let** c: Color = Color.Green;

**+Boolean**: **let** isDone: boolean = false;

**+Turle**: **let** x: [string, number];

x = ["hello", 10];

**+Any**

**+Array**: **let** list: Array<number> = [1, 2, 3];

\_ Hàm:

*function tinhtong(a: number,b: number):number{*

*return a+b;*

*}*

\_ Anonmyous function:

Kiểu 1:

*var hoitham = function(sotuoi: number, hoten: string)*

*{*

*return `chao ${hoten}, minh nam nay ${sotuoi} tuoi.`;*

*}*

*console.log(hoitham(20,"kimhai"));*

Kiểu 2:

*var hoitham02:(sotuoi: number, hoten:string)=> string= function(sotuoi,hoten){*

*return `chao ${hoten}, minh nam nay ${sotuoi} tuoi.`;*

*}*

*console.log(hoitham(20,"kimhai"));*

*Kiểu 3:*

*var hoitham03 = (sotuoi:number,hoten:string):string=>{*

*return `chao ${hoten}, minh nam nay ${sotuoi} tuoi.`;*

*}*

*console.log(hoitham03(30,"kimhai"));*

\_ Class:

TẠO CLASS:

*class nhanvatgame{*

*tennhanvat: string ;*

*slogan: string;*

*mau: number;*

*contructor(tennhanvat:string, slogan: string, mau:number){*

*this.tennhanvat=tennhanvat;*

*this.slogan=slogan;*

*this.mau= mau;*

*}*

*}*

\_Static:

dùng để phân quyền

Giúp ta sử dụng thuộc tính, phương thức mà không cần tạo 1 instance gì cả.

\_ **Kế thừa** trong TypeScript:

WHAT: Kế thừa là thừa hưởng lại các thuộc tính và phương thức của bố.

WHY: Tiết kiệm dung lượng nhớ.

Dễ thay đổi hàng loạt.

Có tính hệ thống hơn

HOW: extends : Khai báo class con.

Super() : hàm sử dụng trong constructor để khỏi phải sử dụng lại code.

Super.function(): Copy nguyên hàm của function bố, sau đó ta định nghĩa thêm các cái mới.

# \_Access Modifier: 3 từ khóa public, private, protected

# \_Accessor: Vừa truy cập được, vừa bảo mật được.

# + getter : Lấy.

# + setter : gán, đặt.

# + có dấu “\_” trước tên phần tử.

# + có 2 hàm SET(), GET(): public get ten() : string return this.\_ten } public set ten(v: string){ this.\_ten = v;

# }

\_ Abtract :

Chỉ dùng để thiết kế Class*. ~~Abstract class Dienthoai{}~~*

Xem class hoạt động thế nào. *~~Var Samsung = new Dienthoai();~~*

Sinh ra là để kế thừa. Không phải để dùng*. Class Android extends Dienthoai{};*

Sinh ra để mô tả tính năng của class, sau đó chính xác hóa nó bằng cách sử dụng lại bản thiết kế abtract

\_ Abtract method:

Là bản thiết kế cho function.

Khi áp dụng abtract vào method thì chỉ ghi kiểu dữ liệu trả về thui. Còn chi tiết của nó sẽ được định nghĩa lại trong Class kế thừa, nếu không sẽ bị báo lỗi ngay.

\_ Generic:

What hiểu đơn giản là cách khai báo hàm nhưng không cần chỉ ra kiểu dữ liệu của đối số và của chính hàm.

Why Hơn hàm bình thường ở chỗ là: vừa có thể nhận giá như hàm bth, vừa vào cả kiểu dl.

How Generic function: Sử dụng dấu <T> để làm kí hiệu generic: *function vd1 <T> (x:T) : T{}*

*Generic Class: class Dienthoai<X,Y,Z>{}*

\_ Interface:

What giống như giao diện web, cho biết chức năng chạy thế nào.

Nhưng chưa triển khai thật

Gần giống abtract , qui mô áp dụng nhỏ hơn.

Why người leader vẽ ra các chức năng chi tiết của các hàm, class

Đảm bảo tính hệ thống cho chương trình.

Nhóm các kiểu dữ liệu với nhau thành 1 loại.

How interface nguoi{

Ten?: string; //có thể có hoặc không

tuoi: number; //ép buộc có

}

xemtt({ten: 'hai', tuoi: 20});

\_ Interface Class:

What Bảng thiết kế cho Class

Nhưng các hàm không cần viết chi tiết.

Chỉ cần khai báo kiểu dữ liệu,

How

interface tuong{

mau:number;

ten:string;

kinang():void;

xemtt():string;

}

class tuongAxtrox implements tuong{};

\_Name space:

What: đặt tên biến mà trùng với từ khóa thì không được.

Name space được sinh ra để giải quyết vấn đề ấy.

Why Đặt tên class mà không bị trùng từ khóa

Quản lý theo một nhóm gọi là module có hệ thống.

How gom hết các class vào bên trong module.

Dùng đến class nào thì sử dụng từ khóa export trước từ khóa đó

Khi gọi thì gọi new tenmodule .tenClass();

*module water{*

*export class string{*

*}*

*}*

*var coca=new water.string();*

*coca.xuatten();*