Cheatography

TypeScript Cheat Sheet
by Greg Finzer (GregFinzer) via cheatography.com/15280/cs/10814/

Types

String	let customerName: string= "John Doe";
Number	let price: number = 19.95;
Boolean	let shipped: boolean = false;
Date	let orderDate: Date = new Date(2017, 2, 9);
Any	let something: any = "Can be anything";
Enum	enum Color {Red, Green, Blue};
Array	let cards: string[] = ['Visa', 'MasterCard'];
Null	let orderId: number = null;
Tuple	let stateTaxRates: [string, number];
Void	function log(msg: string): void { console.log(msg); }
Const	const lives: number = 99;

Classes

```
class OrderLogic {
    constructor(public order: IOrder) { }
    getOrderTotal(): number {
        let sum: number = 0;
        for (let orderDetail of
this.order.orderDetails)
        {
            sum += orderDetail.price;
        }
        return sum;
    }
}
```

Abstract Classes

```
abstract class Person {
  name: string;
  monthlySalary: number;
  monthlyBenefits: number;
  abstract calcSalary(): number;
}
```

Inheritance and Implementing Interfaces

```
interface IGPS {
  getLocation() number;
}
interface ISelfDrive extends IGPS {
  drive(latitude: number, longitude: number,
elevation: number) : void;
}
class Vehicle {
  make: string;
  model: string;
  year: number;
}
class FlyingCar extends Vehicle implements ISelfDrive
{
   hasGps: boolean;
  drive(latitude: number, longitude: number,
elevation: number) {
  }
  getLocation(): number {
  }
}
```

Usage

Installing TypeScript npm	npm install -g typescript
Compiling TypeScript	tsc somefile.ts
TypeScript Docs	TypeScriptLang.org
Type Definition Files	DefinatelyTyped.org

Scope/Modifiers

Public (default)	public firstName: string;
Protected	protected inventory: number;
Private	private outOfStock: boolean;
Read Only	readonly pi: number = 3.14159;
Static	static log(msg: string) { console.log(msg) };

Interfaces

```
interface IOrderDetail {
  productName: string;
  quantity: number;
  price: number;
  orderDate: Date;
  shipped: boolean;
  //Optional
  outOfStock?: boolean;
  //Method
  calcTax: (taxRate: number) => number;
}
```

Optional Parameters

```
class Util {
  log(msg: string, logDate?: Date) {
    if (logDate)
      console.log(logDate + ' ' + msg);
    else
      console.log(new Date() + ' ' + msg);
  }
}
```

Rest Parameters

```
class Order {
  addOrderDetails(...orderDetails: IOrderDetail[]) {
  }
}
```

Namespaces

```
namespace AcmeCorp.Logging {
  export class Logger {
      static log(msg: string) : void {
        console.log(msg);
      };
  }
}
/// <reference path="AcmeCorp.Logging.ts" />
//Alias
import logger = AcmeCorp.Logging.Logger;
namespace AcmeCorp.OnlineStore {
  class OrderLogic {
    calcOrder(): number {
        logger.log("calculating order");
        return 0;
    }
  }
}
```