**TypeScript**

**General**

1. **Static methods or variables** are not accessible inside function as well as we can’t use static method/ variables in class functions with **‘this’ keyword**. Static property is not available in the class instance.
2. **This** keyword points to the **current instance** we are working with.
3. **Static** properties are **detached from instances**. Use class name to utilize static properties.
4. Marking a method **abstract** will cause its implementation to be compulsory in every child and also must **mark the class** **with abstract keyword.** 
   1. E.g., abstract describe(department: string) : void;
   2. No function body no nothing. Now this describe method is compulsory for all the sub classes.
   3. Sub classes does not have to mark with abstract keyword but the parent class should.
5. Index properties (also known as **index signatures**) in TypeScript allow you to define the shape of an object when the names of its properties are not known in advance but the types of their values are known.
   1. Syntax: type TypeName = { [key: KeyType]: ValueType; };
   2. Example:
      1. type MixedDictionary = {

[key: string]: number;

id: number; // Specific properties can coexist with index signatures

};

const mixed: MixedDictionary = {

id: 101,

age: 25,

};

* 1. You can add much as properties you want just make sure if **matches the *key* and *value* Type**.

1. The **nullish coalescing operator (??)** is used to provide a default value when the left-hand side is null or undefined.
   1. Syntax: const result = value ?? defaultValue;
2. || treats falsy values (0, false, "") as equivalent to null or undefined. But ‘??’ does not.
3. In TypeScript, you can attach **multiple decorators** to a single class, and they are evaluated **in reverse order** (from bottom to top).
4. Decorators are more famous in angular than react.
5. Decorators are activated when you define a class, no need to instantiate a class to call decorators.