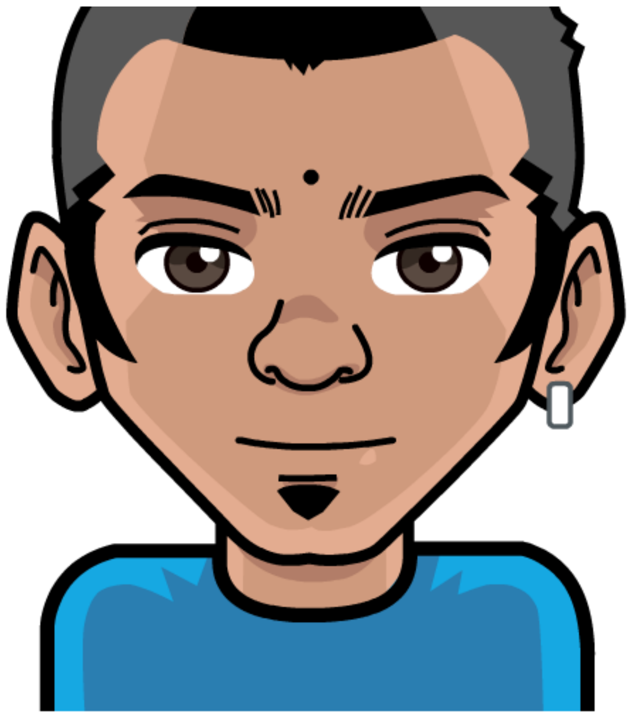
JUST ENOUGH TYPESCRIPT



RAJU GANDHI

@LOOSELYTYPED

CTO - INTEGRALLIS SOFTWARE

TYPESCRIPT?

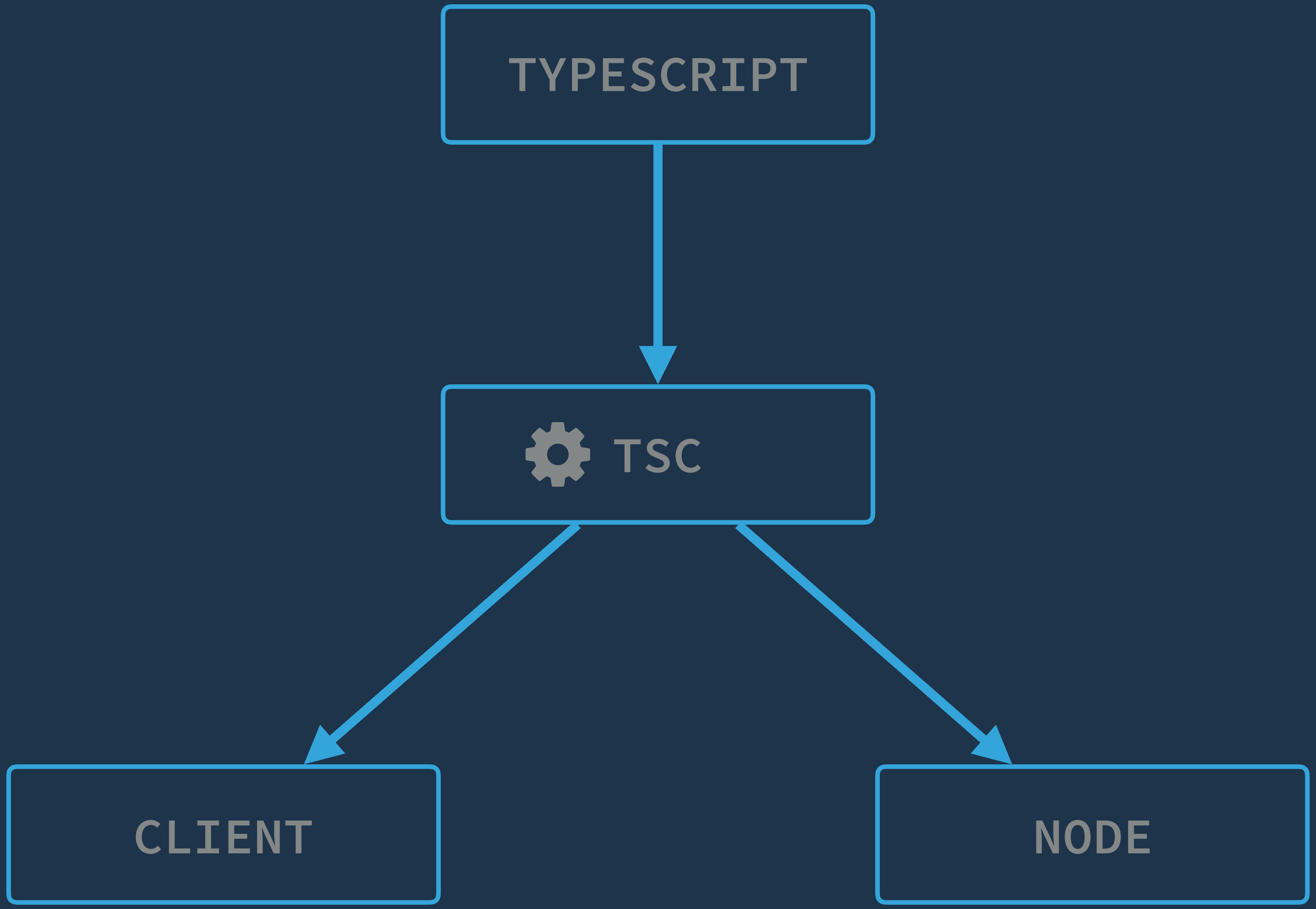
TYPESCRIPT

- MICROSOFT
- SUPERSET OF JAVASCRIPT …
- WITH OPTIONAL TYPING

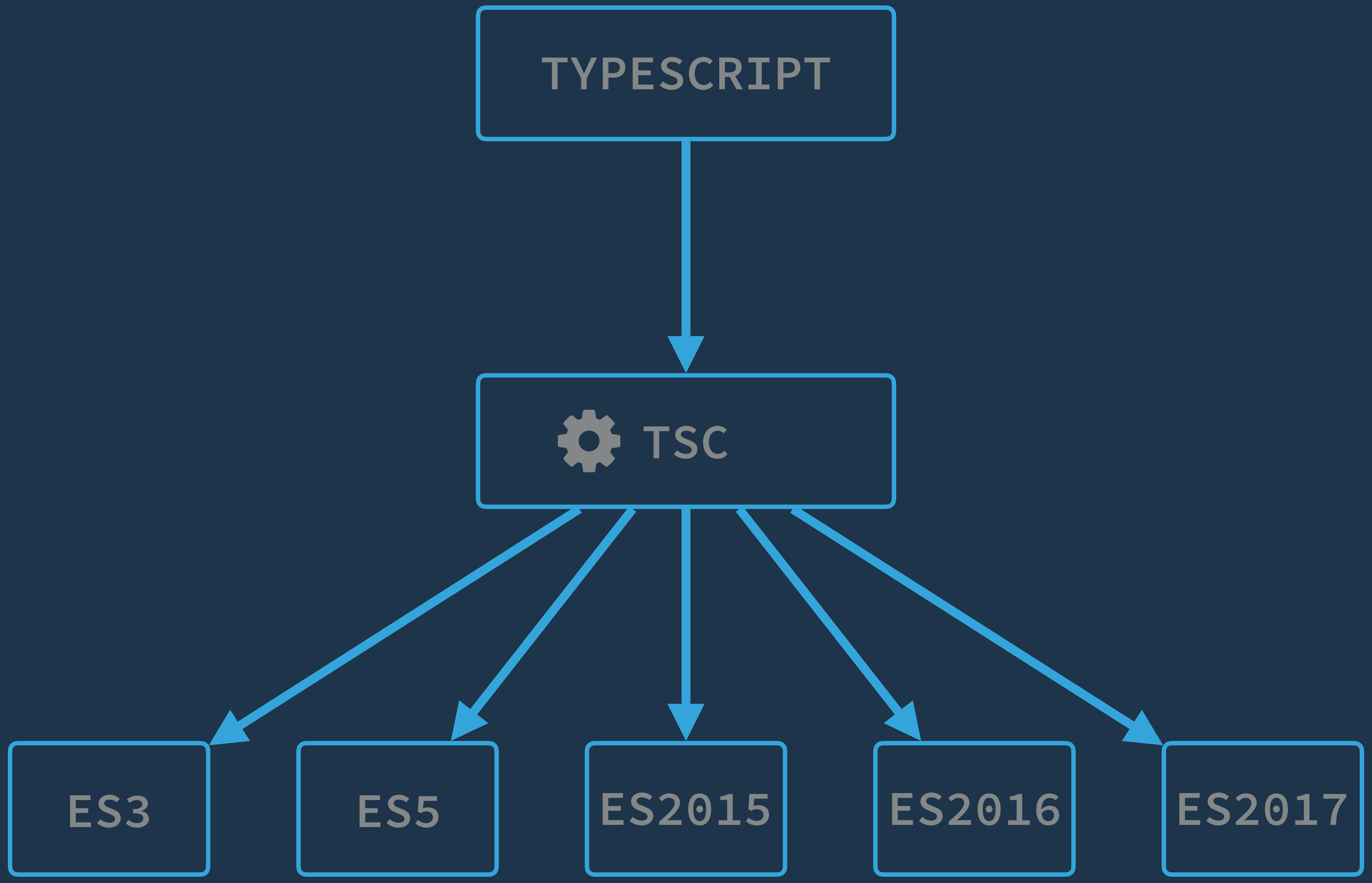
What's
NEXT

SETUP

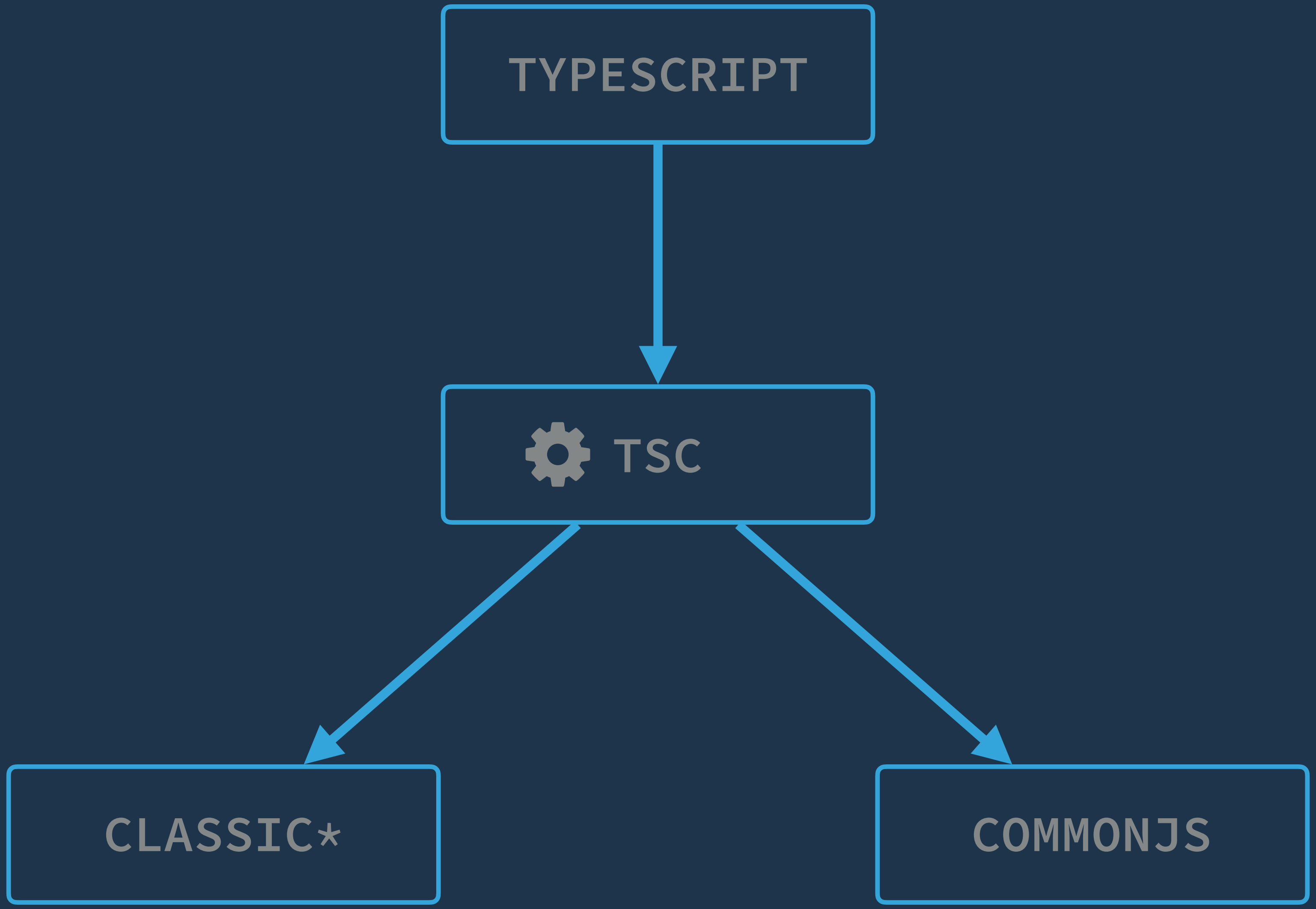
TYPE



TARGET



MODULE RESOLUTION



TSCONFIG.JSON

SAMPLE

```json
{
  "compileOnSave": true,
  "compilerOptions": {
    "emitDecoratorMetadata": true,
    "experimentalDecorators": true,
    "module": "es2015",
    "moduleResolution": "node",
    "noImplicitAny": true,
    "outDir": "dist",
    "strict": true,
    "target": "es6"
  },
  "include": [
    "src/**/*.ts"
  ],
  "exclude": [
    "node_modules"
  ]
}
```

ANGULAR DEFAULT

```json
{
  "compilerOptions": {
    "baseUrl": "",
    "declaration": false,
    "emitDecoratorMetadata": true,
    "experimentalDecorators": true,
    "lib": [
      "es2016",
      "dom"
    ],
    "mapRoot": "./",
    "module": "es2015",
    "moduleResolution": "node",
    "outDir": "../dist/out-tsc",
    "sourceMap": true,
    "target": "es5",
    "typeRoots": [
      "../node_modules/@types"
    ]
  }
}
```

ADDITIONAL (DEV) OPTIONS

```
{
  "compileOnSave": true, //Needs editor support
  "compilerOptions": {
    "alwaysStrict": true,
    "noFallthroughCasesInSwitch": true,
    "noImplicitReturns": true,
    "noImplicitThis": true,
    "noUnusedLocals": true,
    "noUnusedParameters": true,
    "sourceMap": true,
    "strictNullChecks": true,
    "noImplicitAny": true,
    "pretty": true, //Stylize errors and messages
    "strict": true
  }
}
```

INIT

```
mkdir project_name && cd project_name
tsc --init
```

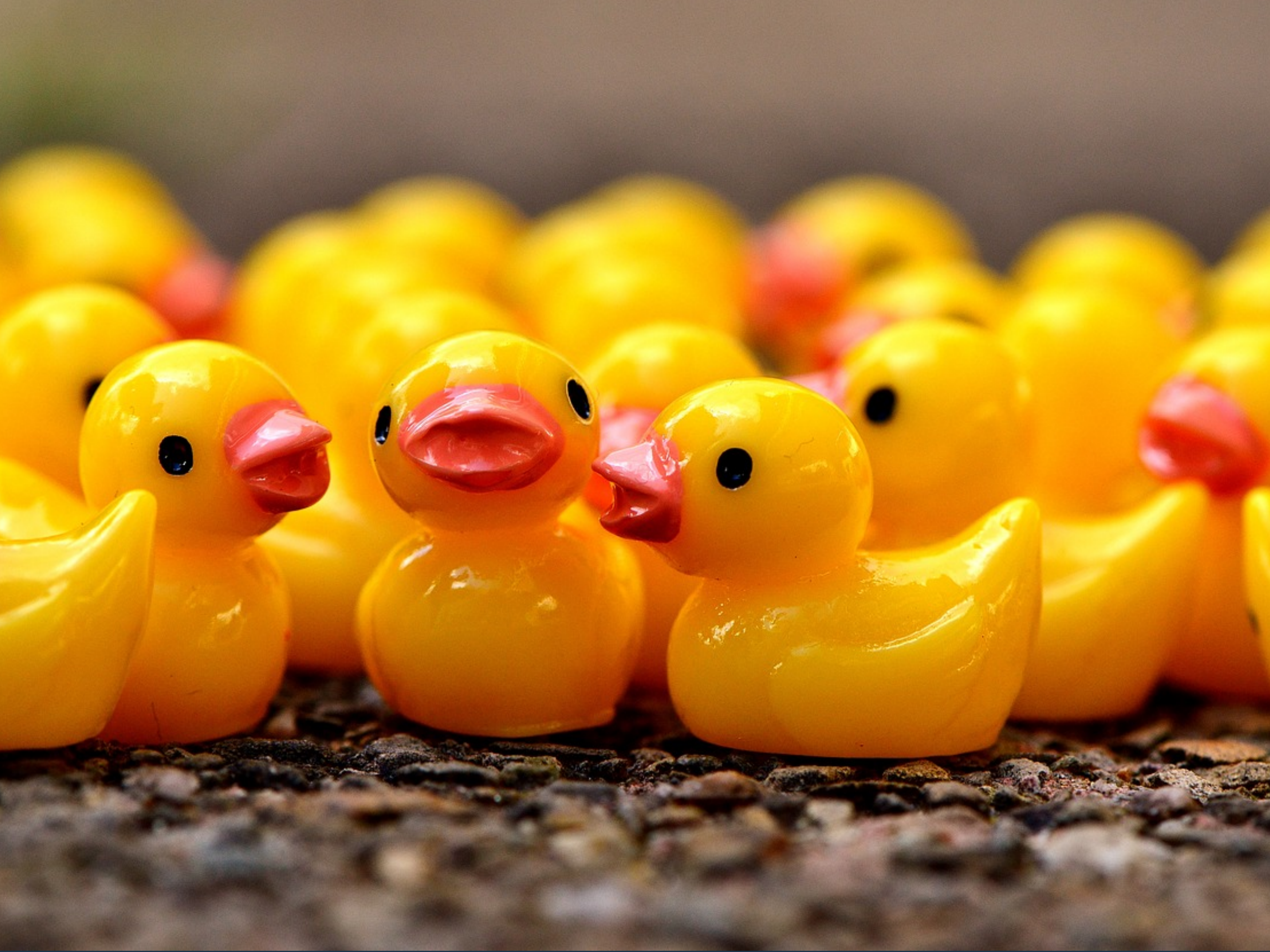
tsconfig schema link

FEATURES

TS $\supseteq$ JS

HAS IT ALL ...

- FAT ARROW FUNCTION
- CLASSES
- MODULE SUPPORT
- PROMISE
- ITERATORS/GENERATORS
- ASYNC/AWAIT



```typescript
interface Point {
  readonly x: number;
  readonly y: number;
  readonly z?: number;
}

const describePoint = (p: Point): string => {
  return `The point coords are x:${p.x} y:${p.y} z:${p.z}`;
}

const myPoint = {
  x: 10,
  y: 15
};

console.log(describePoint(myPoint));
```

NOTE

- USE CONST FOR VARIABLES, READONLY FOR PROPERTIES

BASIC TYPING

```typescript
let aString: string;
aString = "TypeScript Rocks!";
// s = 10; // ERROR

let anotherString: string = "This is inferred";
```

ACCESS MODIFIERS

```
class ModifierDemo {
  public name = "super";
  protected id = 1;
  private password = "password";

  public toString() {
    return `I can see all -- ${this.name} ${this.id} ${this.password}`
  }
}

const superInstance = new ModifierDemo();
superInstance.name;
// superInstance.id; // ERROR
// superInstance.password; //ERROR
console.log(superInstance.toString());
// I can see all -- super 1 password
```

```
class ModifierDemoSub extends ModifierDemo {
 public myToString() {
    // CANNOT see password
    return `Subclass can see -- ${this.name} ${this.id}`
  }
}

const subInstance = new ModifierDemoSub();
subInstance.name;
// subInstance.id; // ERROR
// subInstance.password; // ERROR
console.log(subInstance.toString()); // WORKS
// I can see all -- super 1 password
console.log(subInstance.myToString()); // WORKS
// Subclass can see -- super 1
```

FUNCTION TYPES

```
type CallBack = (args: any[]) => void;

const click = (c: CallBack) : void => {
  // do something here
}
```

ENUMS

```typescript
type CallBack = (args: any[]) => void;

const click = (c: CallBack) : void => {
  // do something here
}
```

GENERICS

```
class Stack<T> {
  private items: T[];

  push = (item: T): T => {
    this.items.push(item);
    return item;
  }

  pop = (): T | undefined => this.items.shift();
}

const stack = new Stack<string>();

stack.push('typescript');
stack.push('javascript');
// stack.push(true);  // ERROR
```

```typescript
const tap = <T>(f: (arg: any) => any, a: T): T => {
  f(a);
  return a;
};
```

ADVANCED

FUNCTION OVERLOADING

```
class Border {
  top: number;
  right?: number;
  bottom?: number;
  left?: number;

  static border(all: number): Border;
  static border(topAndBottom: number, leftAndRight: number): Border;
  static border(top: number, right: number, bottom: number, left: number): Border;
  static border(a: number, b?: number, c?: number, d?: number): Border {
    if(!b) b = a;
    if(!c) c = a;
    if(!d) d = b;
    return {
      top: a,
      right: b,
      bottom: c,
      left: d
    };
  }
}
```

NOTE

- DECLARE MORE SPECIFIC SIGNATURES <u>AFTER</u> LESS SPECIFIC ONES

NULLABLE TYPE

```
// with "strictNullChecks": true
let supplied: string;
// pin = undefined; // ERROR

let optional: string | undefined;
optional = 'a value';
optional = undefined;

type User = {
  firstName: string,
  lastName: string,
  middleInitial: string | undefined,
};
```

```typescript
function processPin(pin: string | undefined): string {
  // return p.toUpperCase(); ERROR
  return pin ? pin.toUpperCase() : 'reset';
}
```

UNION TYPE

```
class Apple {
  getColor(): string {
    return "red";
  }
  isRipe(): boolean {
    return true;
  }
}

class Plum {
  getFlavor(): string {
    return "tart";
  }
  isRipe(): boolean {
    return false;
  }
}

const interrogateFruit = (fruit: Apple | Plum): boolean => {
  // fruit.getColor(); // ERROR
  // fruit.getFlavor(); // ERROR
  return fruit.isRipe();
};
```

```
let fruits: (Apple | Plum)[] = [
  new Plum(), new Apple(), new Apple(), new Plum()
];

fruits.filter(f => f.isRipe());
```

LITERAL TYPES

```
// number literal
let port: 80 | 443;
// if(port === 100) { } // ERROR

// string literal
let scheme: 'http' | 'https';

// function using literal types
const getPort = (scheme: "http" | "https"): 80 | 443 => {
  switch (scheme) {
    case "http":
      return 80;
    case "https":
      return 443;
  }
}
```

```typescript
enum UserStatus {
  ACTIVE,
  INACTIVE,
}

function getProfile(status: UserStatus.ACTIVE): 'exists';
function getProfile(status: UserStatus.INACTIVE): 'does not exist';
function getProfile(status: UserStatus): 'exists' | 'does not exist' {
  switch (status) {
    case UserStatus.ACTIVE:
      return 'exists';
    case UserStatus.INACTIVE:
      return 'does not exist';
  }
}
```

DECORATORS

METHOD DECORATOR

```
function timed(target: any,
               propertyKey: string,
               descriptor: PropertyDescriptor) {
  const original = descriptor.value;

  descriptor.value = function (...args: any[]) {
    const start = Date.now();
    var result = original.apply(this, args);
    console.log(`${propertyKey} took ${Date.now() - start} ms`);
    return result;
  };
};

class C {
  @timed
  double(n: number) {
    return n * 2;
  }
}
```

```
function timedFactory(prefix = '') {
  return function timed(target: any,
                        propertyKey: string,
                        descriptor: PropertyDescriptor) {
    const original = descriptor.value;

    descriptor.value = function (...args: any[]) {
      const start = Date.now();
      var result = original.apply(this, args);
      console.log(`${prefix}: ${propertyKey} took ${Date.now() - start} ms`);
      return result;
    };
  };
}

class D {
  @timedFactory('class D')
  foo(n: number) {
    return n * 2;
  }
}

const d = new D();
d.foo(3);
```

DECORATORS

- DATA ABOUT DATA
- 4 KINDS (CLASS, METHOD, PROPERTY, PARAM)
- DECORATOR FACTORIES

TOOLING

TOOLING

- VISUAL STUDIO CODE
- TSLINT
- DEFINITELY TYPED
- ESLINT

TSLINT

```
npm install tslint tslint-eslint-rules --save-dev;
tslint --init;

// sample tslint.config file
{
  "defaultSeverity": "error",
  "jsRules": {},
  "rulesDirectory": [],
  "extends": [
    "tslint-eslint-rules"
  ],
  "rules": {
    "no-console": [true, "log"],
    "no-duplicate-imports": true,
    "no-duplicate-variable": true,
    "no-var-keyword": true,
    "semicolon": [true],
    "variable-name": [true,"ban-keywords"],
    "no-inner-declarations": [true,”function"]
  }
}

// sample package.json scripts
{
  "scripts": { "lint": "tslint -c tslint.json 'src/**/*.ts' && exit 1” },
}
```

@TYPES

```
npm install @types/node @types/express @types/debug --save-dev
npm install @types/body-parser @types/morgan --save-dev
```

RESOURCES

TYPESCRIPT DOCS

MARIUS SCHULZ BLOG

CREDITS

THEME - HTTPS://SPEAKERDECK.COM/PHILHAWKSWORTH/EXCESSIVE-ENHANCEMENT-GOTHAMJS

IMAGES

- WHAT'S NEXT - HTTPS://PIXABAY.COM/EN/BOARD-SCHOOL-IMMEDIATELY-SOON-1647323/
- DUCKY - HTTPS://SPEAKERDECK.COM/PHILHAWKSWORTH/EXCESSIVE-ENHANCEMENT-GOTHAMJS

THANKS