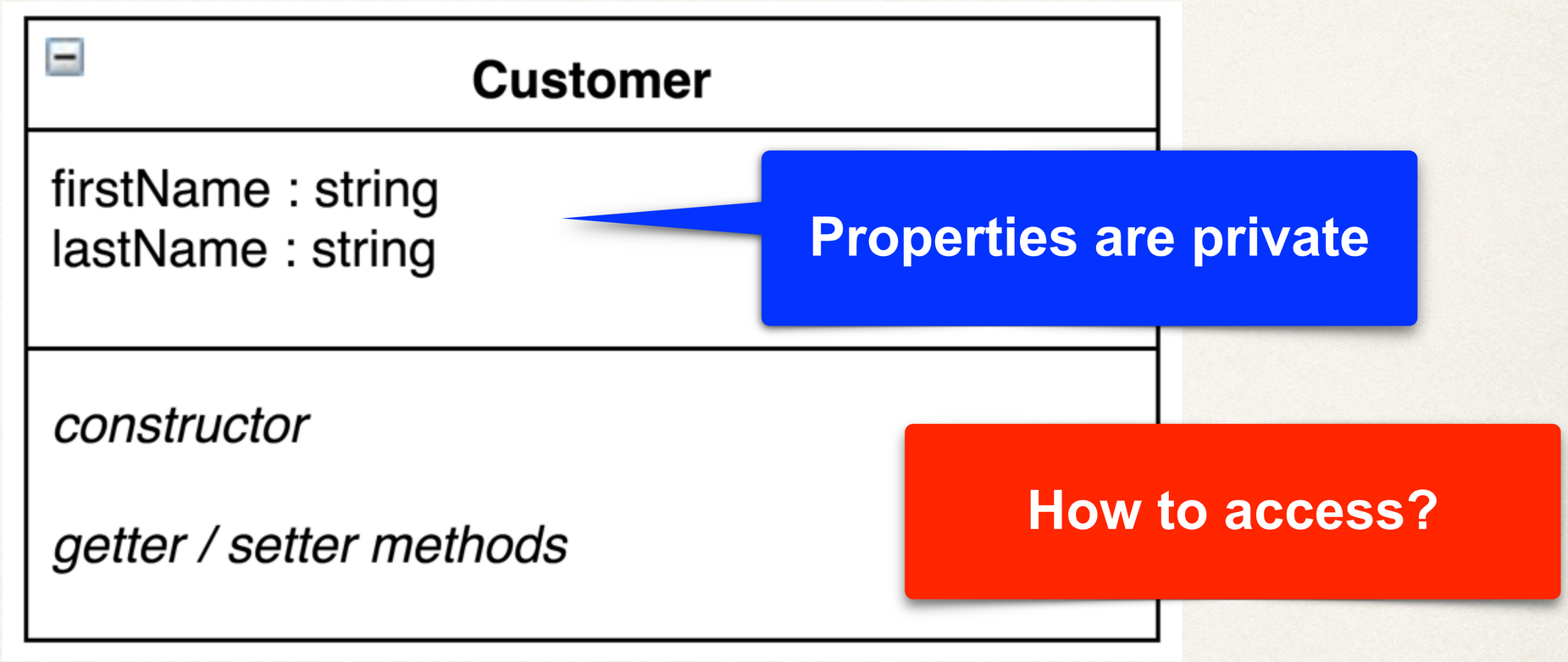
luv2code

TypeScript - Accessors

Customer Class



Getter / Setter Methods

- Since our properties are private, we need a way to access them

- We can create traditional methods as in other OO languages:
  - Define getter/setter methods

Getter / Setter Methods

File: Customer.ts

```
class Custome

    private firstNa e: string;
    private lastNam : string;

    public get irstName(): string {
        return this.firstName;
    }

    public setFirstName(theFirst: string): void {
        this.firstName = theFirst;
    }
}
```

Method name

Method name

Return type

Param type

Return type

```
// now let's use it
let myCustomer = new Customer("Martin", "Dixon");

myCustomer.setFirstName("Greg");
console.log(myCustomer.getFirstName());
```

TypeScript: Accessors - Get / Set

- TypeScript also offers an alternate syntax
- Define special: get / set methods
- Known as **Accessors**

TypeScript: Accessors - G

Code outside of the class is NOT accessing internal properties directly

Calls the "get" accessor since we are accessing a value

Can give any property name

Note the syntax:
get <space> property()

Note the syntax:
set <space>

Calls the "set" accessor since we are assigning a value

Just a convention

No return type … not even "void"

```
    private _lastName: st
    … …

    public get firstName
        return this._firstName;
    }

    public set firstName(value: string) {
        this._firstName = value;
    }
}
```

```
Dixon");
```

TypeScript: Accessors - Get / Set

Can give any internal name

File: Customer.ts

```
class Customer {

    private x: string;
    private y: string;
    … …

    public get firstName(): string {
        return this.x;
    }

    public set firstName(value: string) {
        this.x = value;
    }
}
```

```
// now let's use it
let myCustomer = new Customer("Martin", "Dixon");

myCustomer.firstName = "Susan";
console.log(myCustomer.firstName);
```

The public get/set accessors are still called accordingly

TypeScript: Accessors - Get / Set

Removed “public” on the accessors.
If no access modifier given, “public” by default

File: Cust

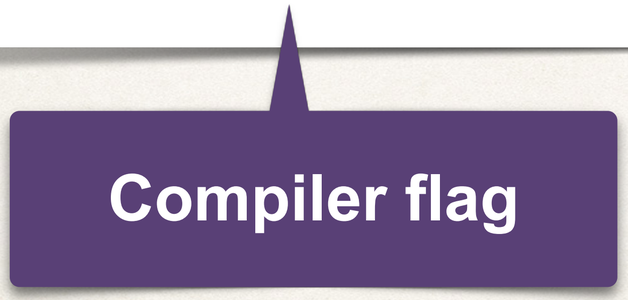
```
class Customer {

    private _firstName: string;
    private _lastName: string;
    … …

    get firstName(): string {
        return this._firstName;
    }

    set firstName(value: string) {
        this._firstName = value;
    }
}
```

Renamed to the original names from previous slides

Compiler flag

- The get/set accessors feature is only supported in ES5 and higher
- You have to set a compiler flag in order to compile the code

```
C:\> tsc --target ES5 --noEmitOnError Customer.ts
```



Problem with too many compiler flag

- You may have noticed, that we have a lot of compiler flags
  - Too much stuff to remember … easy to forget
- Wouldn't it be great to set this up in a config file?
- TypeScript has a solution: **tsconfig.json** file

tsconfig.json

- **tsconfig.json** file defines compiler options and project settings
- Place this file in the root of your project directory

File: tsconfig.json

```
{
  "compilerOptions": {
    "noEmitOnError": true,
    "target": "es5"
  }
}
```

tsconfig.json

- You can also generate a template for this file

```
C:\> tsc --init
```

Generates a default tsconfig.json file

- Then edit the **tsconfig.json** accordingly for your project requirements

Compiling your Project

- Once your project has a **tsconfig.json** file, then you can compile with

```
C:\> tsc
```

No need to give names of TypeScript files.
By default, will compile all *.ts files

www.luv2code.com/tsconfig-docs

TypeScript: Accessors - Get / Set

File: Customer.ts

```
class Customer {

    private _firstName: string;
    private _lastName: string;
    … …

    get firstName(): string {
        return this._firstName;
    }

    set firstName(value: string) {
        this._firstName = value;
    }
}
```

```
// now let's use it
let myCustomer = new Customer("Martin", "Dixon");

myCustomer.firstName = "Susan";
console.log(myCustomer.firstName);
```