

Singleton

Builder design pattern

1) class : lot of attributes

```
Student {  
    id  
    name  
    batch  
    psp  
    currentMentor  
    attendance  
    course  
    list < Electives >  
}
```

```
HolidayPackage {  
    days  
    nights  
    country  
    city  
    hotelKind  
    flight  
    cab  
}
```

```
Car {  
    seats  
    Engine  
    type  
    transmission  
    fuel  
}
```

```
House {  
    rooms  
    isSwim  
    isWooden  
}
```

Student s = new Student(—, —, —, —, —, —);

Student(—, —)

Student(—, —, —, —)

⋮

getters and setters

←
s1 = Student(); ✓
s1.setAttr1();
s1.setAttr2();
⋮
⋮

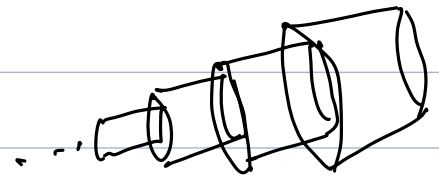
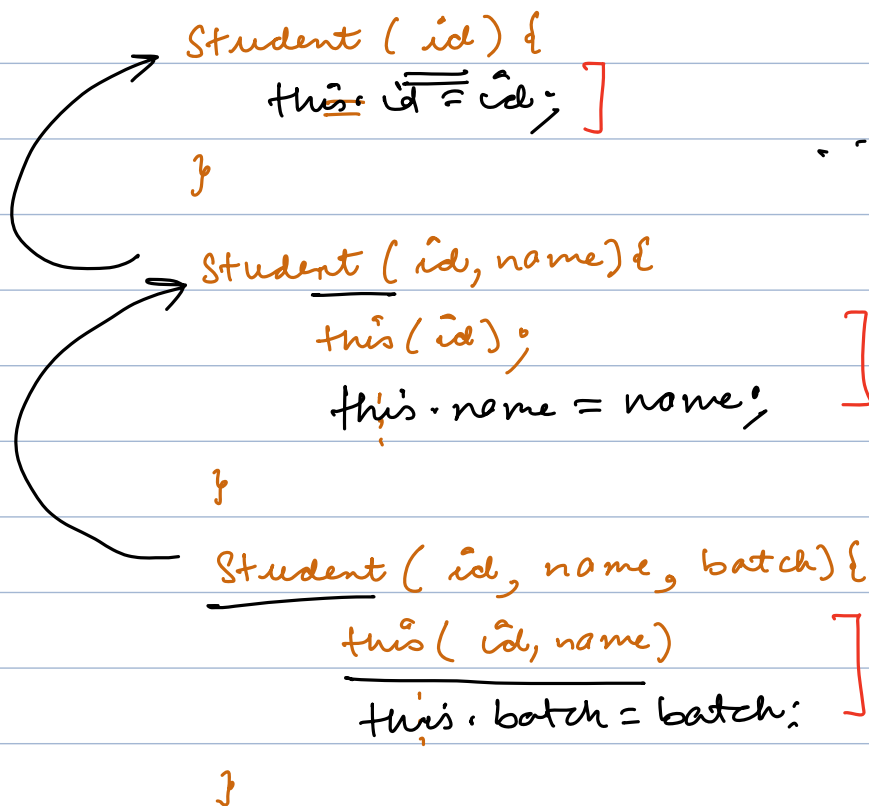
attr1
→ validation

Validation's before
creating the obj

{ gradYear > 2024 .
age < 20

Telescoping constructors

```
Student ( id ) {  
    this.id = id;  
}  
  
Student ( id, name ) {  
    this(id);  
    this.name = name;  
}  
  
Student ( id, name, batch ) {  
    this ( id, name )  
    this.batch = batch;  
}
```



Student ()

↑ don't want to
pass a lot of
attributes

Map < String, Object > m;

m.add("id", 1);

m.add("name", "MOHIT");

m.add("age", 25);

↓
{ attributes → values }

↓
HM

what if the client
passes a double?

what if "age"?

{ - something like Map
+
type check on values
* proper keys

↓
class

```
class Builder {
```

// this will contain
the exact same
attributes as student
or lesser based on
what clients need to
send.

```
}
```

```
Builder b = new Builder();  
b.setAge();  
b.setGradYear();
```

```
Student s = new Student(b);
```

↑
validations

Student (— , — , — , —);

10 files

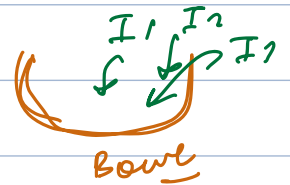
Student (—)

obj → add a new

```

Builder b = new Builder();
// Builder b = Student.getBuilder();
// { b.setAge(29);
//   b.setBatch("Apr23");
//   b.setGradYear(2024);

```



```

Student s = Student.getBuilder()
    .setAge(29)
    .setBatch("Apr23")
    .setGradYear(2024)
    .build();
// Student s = new Student(b);

```

} client

build() {
 new Student (builder this);

Break:

10:35
 |
Break

Friday X
 ↓
Wed

Why Builder?

→ (HM) → class Builder

