

Object-oriented Concepts in JavaScript

Course Introduction



Jim Cooper

Software Engineer

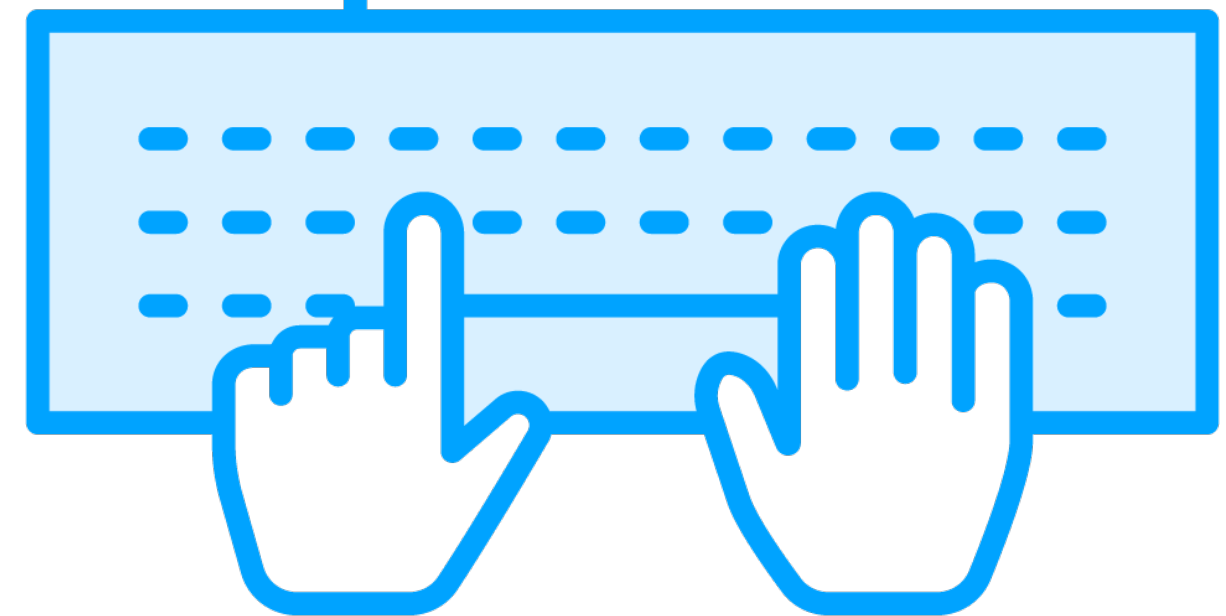
@jimthecoop | jcoop.io



Demonstration focused



Hands-on



```
class Person {  
  #x = [];  
  
  constructor(firstName, lastName) {  
    this.firstName = firstName;  
    this.lastName = lastName;  
  }  
}
```

```
class Student extends Person {  
  #enrolledCourses = [];  
  
  constructor(firstName, lastName) {  
    super(firstName, lastName);  
  }  
  
  get enrolledCourses() {  
    return [...this.#enrolledCourses];  
  }  
  
  ...  
}
```

Build OO Applications with **JavaScript Classes**



**JavaScript is a
Loosely-typed,
Prototype-based
Language**



Agenda

What Prototypes Are

JavaScript Prototypes

Constructor functions

JavaScript Classes



What is Object-oriented Programming?



What is Object-oriented Programming?



Object-oriented Concepts

Encapsulation

Composition

```
let person = {  
  firstName: 'Jim',  
  lastName: 'Cooper',  
  age: 27,  
  isAdult() {  
    return this.age > 18  
  }  
}
```

Person

firstName
lastName

Vehicle

make
model

start()



```
class Person {  
  firstName = '';  
  lastName = '';  
  isAdult() { ... }  
}
```

```
let jim = new Person('Jim', 'Cooper');  
let kris = new Person('Kris', 'Cooper');
```



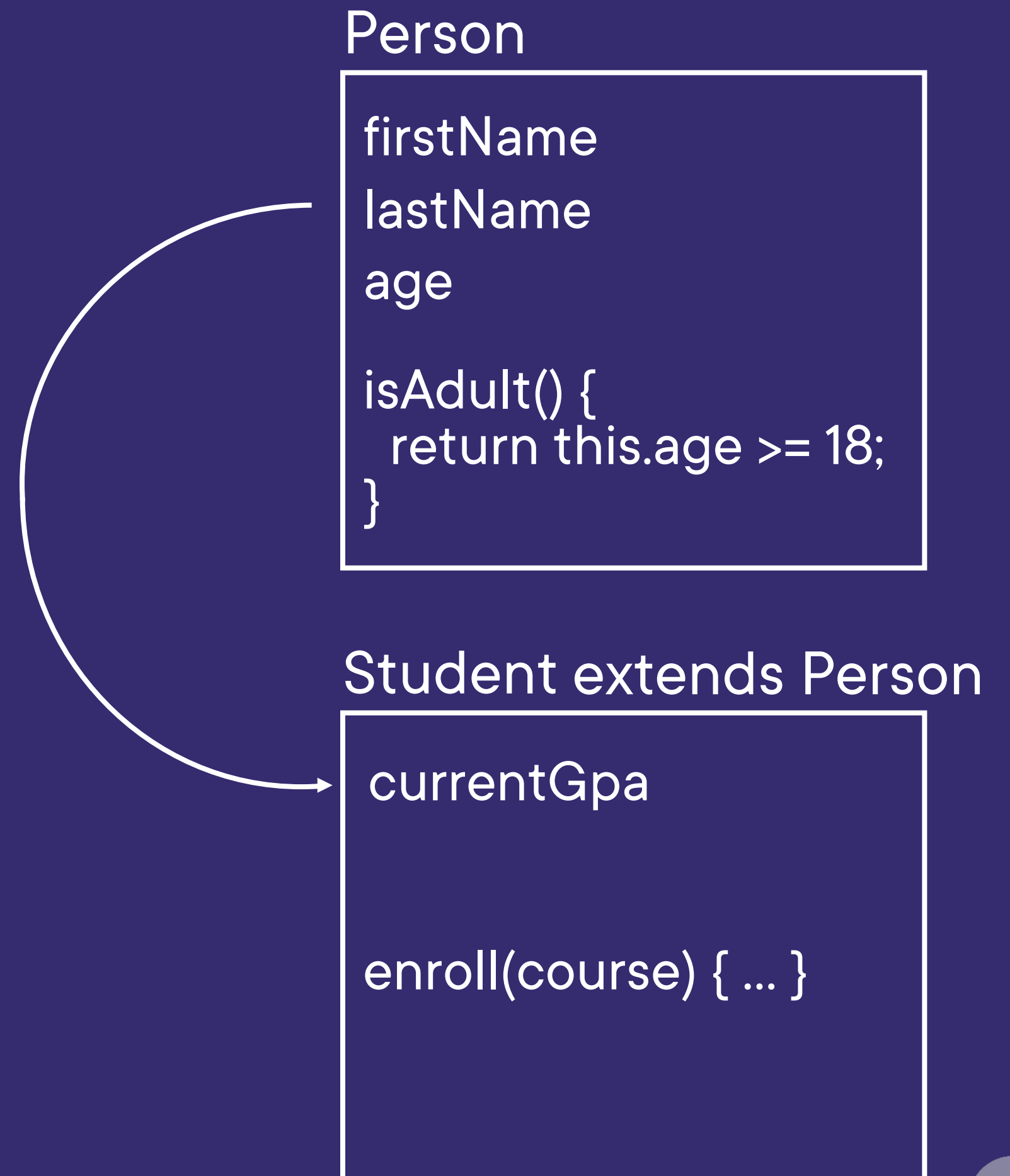
Object-oriented Concepts

Encapsulation

Composition

Inheritance

Polymorphism



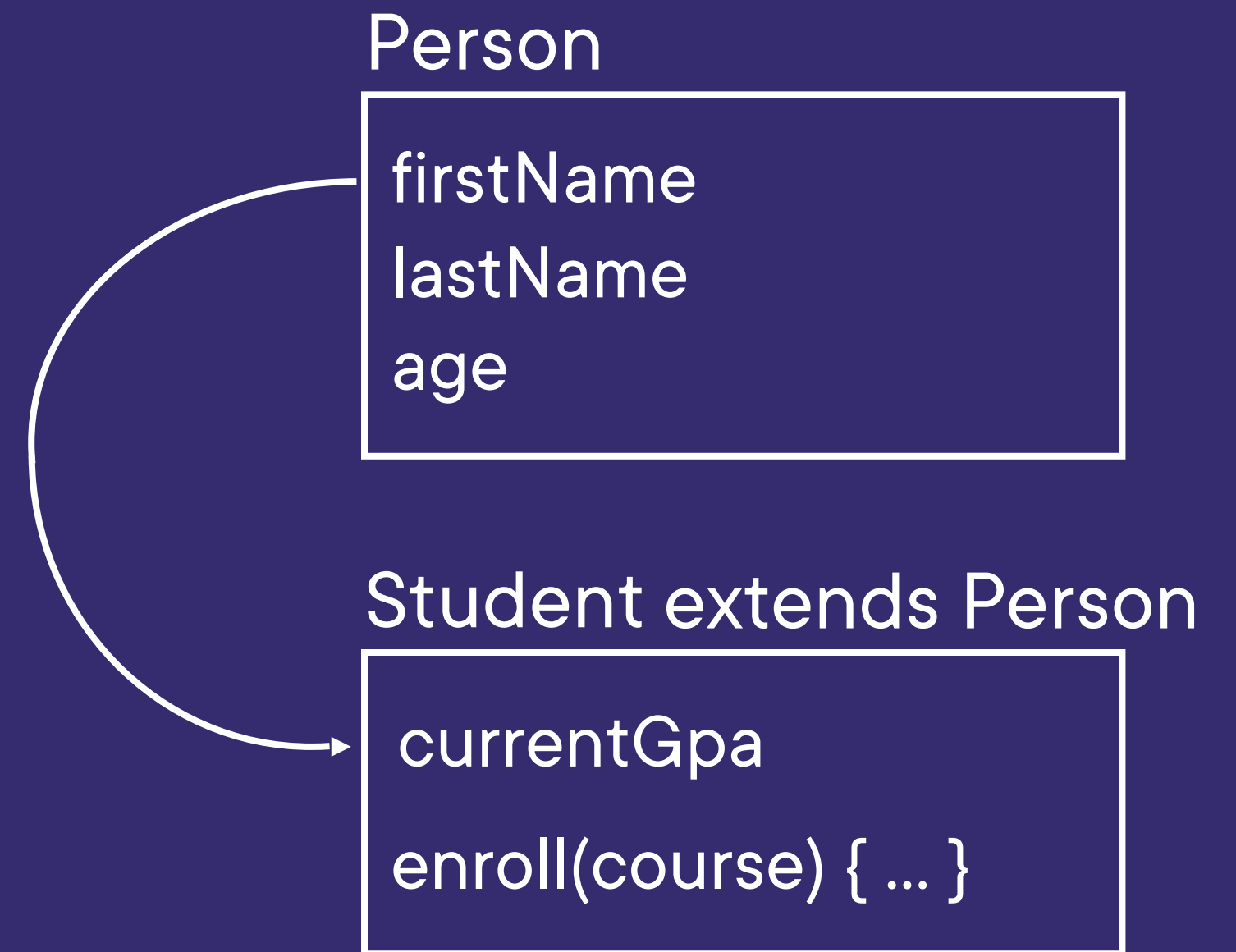
Object-oriented Concepts

Encapsulation

Composition

Inheritance

Polymorphism



```
function getFullName(person) {  
    return person.firstName + person.lastName;  
}
```

```
getFullName( myStudent );
```



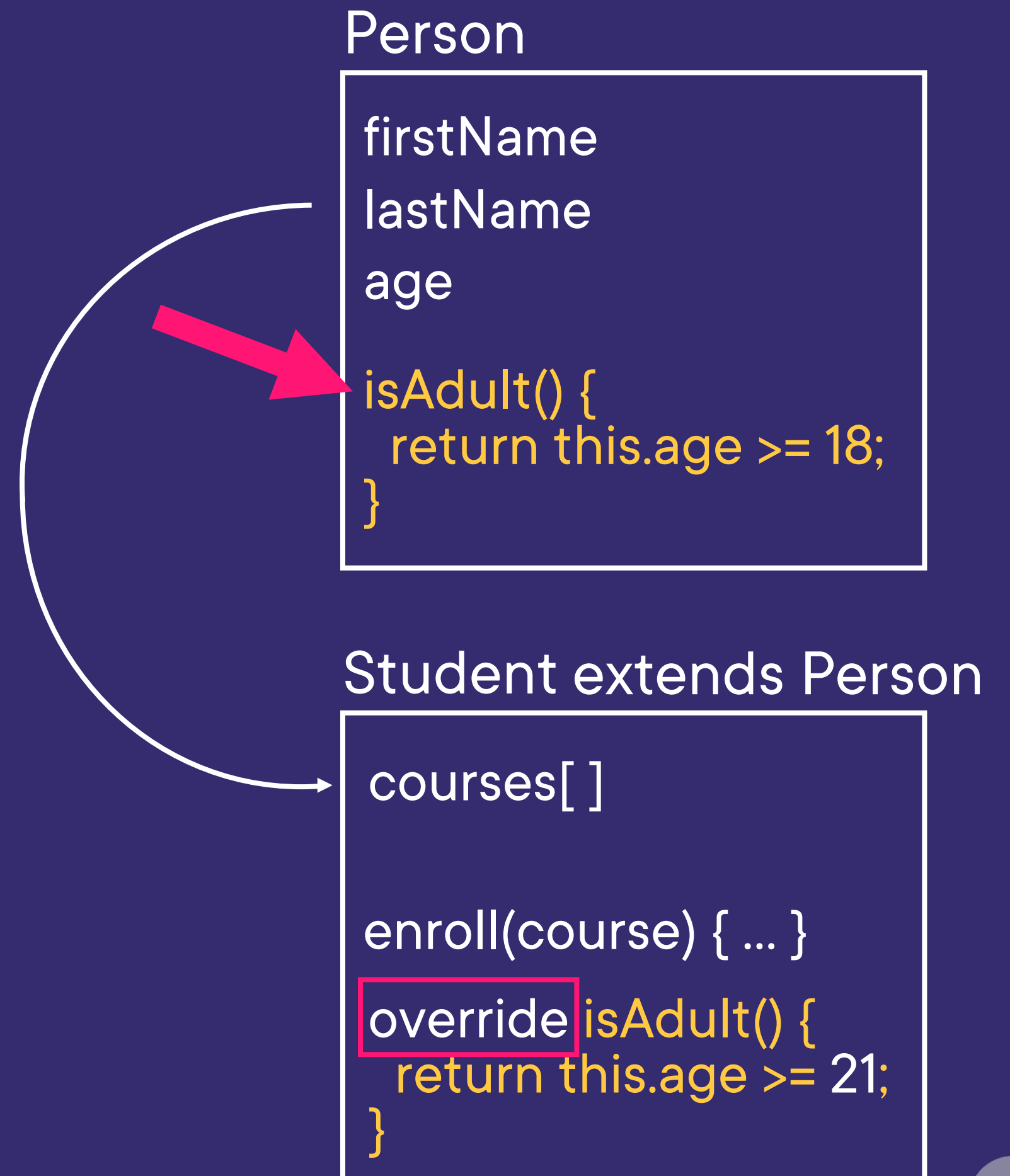
Object-oriented Concepts

Encapsulation

Composition

Inheritance

Polymorphism



**JavaScript is a
Loosely-typed,
Prototype-based
Language**



Traditional (Strongly-typed)

```
int age = 5;
```

integer

```
age = 'five'; // compile error
```

JavaScript (Loosely-typed)

```
let age = 5;
```

number

```
age = 'five';
```

string



Traditional (Strongly-typed)



```
function getFullName(Person person) {  
    return person.fName + person.lName;  
}
```

```
getFullName(student);
```

JavaScript (Loosely-typed)



```
function getFullName(person) {  
    return person.fName + person.lName;  
}
```

```
getFullName(student);
```



Traditional (Strongly-typed)

```
class Person() {  
    ...  
}
```

```
var jim = new Person();
```

Person

JavaScript (Loosely-typed)

```
class Person() {  
    ...  
}
```

```
let jim = new Person();
```

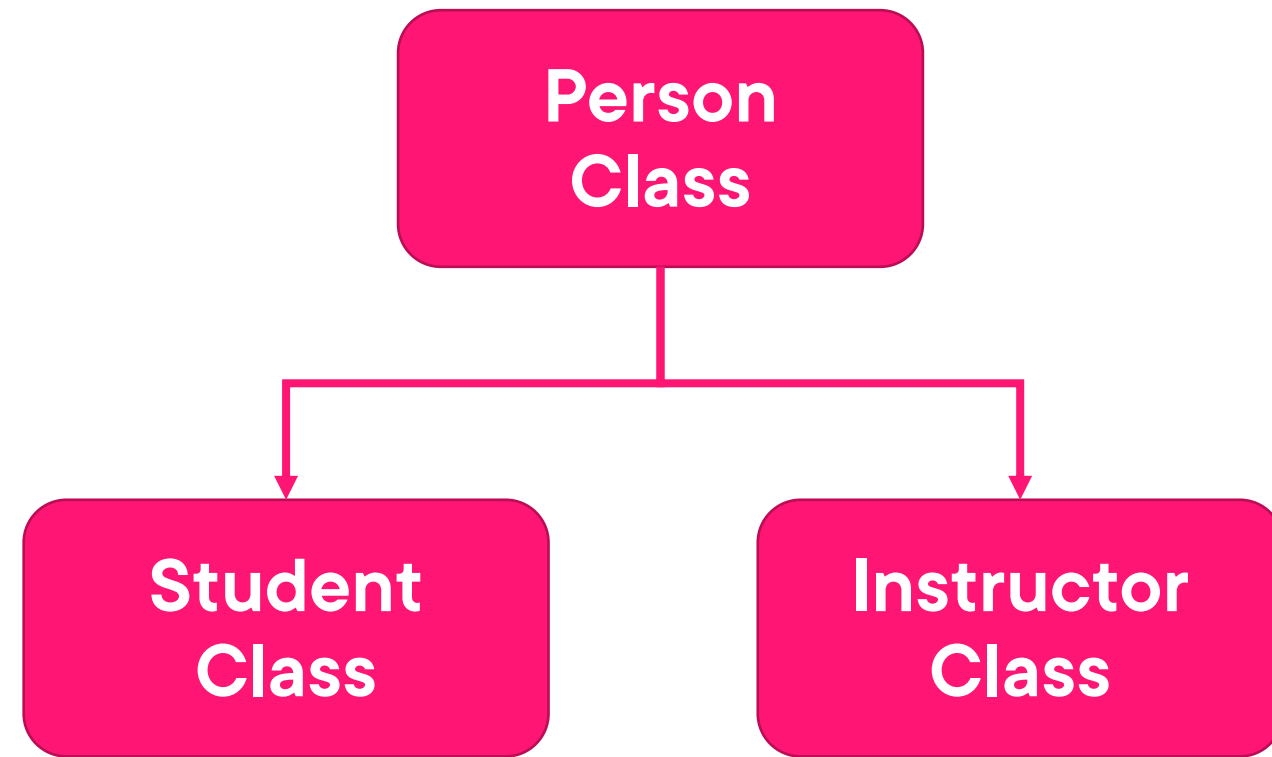
object



**JavaScript is a
Loosely-typed,
Prototype-based
Language**

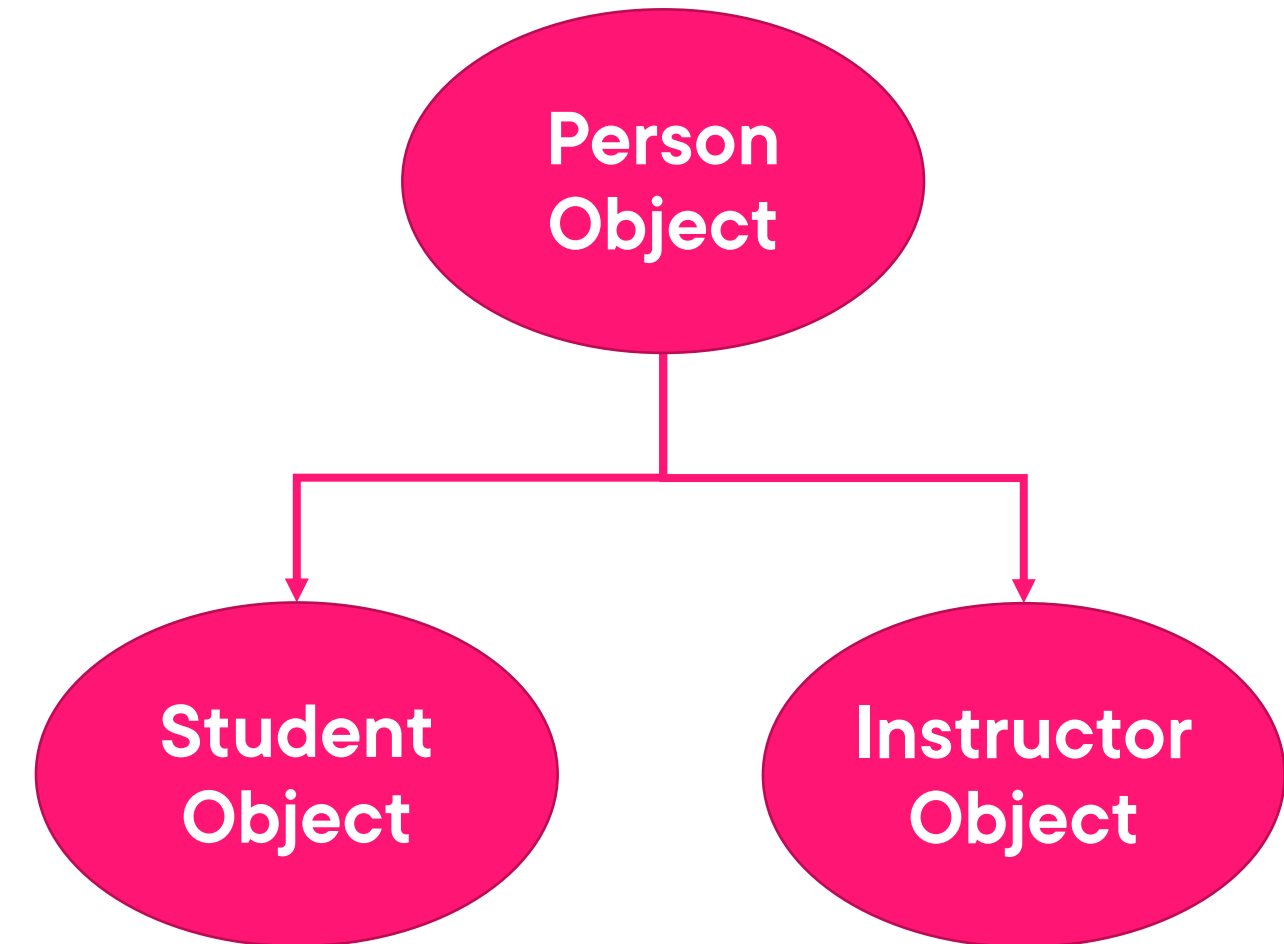


Classes



```
class Person { ... }  
class Student extends Person { ... }  
class Instructor extends Person { ... }  
  
let student = new Student(...);
```

Prototypes



```
let person = {...};  
let student = { prototype: person };  
let instructor = { prototype: person };
```