The JavaScript Language (Part 2)

Robert M. Dondero, Ph.D.
Princeton University

Objectives

- We will cover:
  - A subset of JavaScript…
  - That is appropriate for COS 333…
  - Through example programs

Agenda

- **Modules**
- Objects

Modules

- Kinds of JavaScript modules
  - **ES6** modules
    - Used in (recent) browsers
    - Not used by Node.js
    - Shown later
  - **Node.js** modules
    - Used by Node.js
    - Not used by browsers
    - Shown now…

Modules

- See **euclid2.js** and **euclidclient2.js**

```
$ node euclidclient2.js
Enter the first integer:
8
Enter the second integer:
12
gcd: 4
lcm: 24
$
```

Modules

- See **euclid3.js** and **euclidclient3.js**

```
$ node euclidclient3.js
Enter the first integer:
8
Enter the second integer:
12
gcd: 4
lcm: 24
$
```

Agenda

- Modules
- **Objects**

Objects

- Object definition

```
let someobj = {
    property1: value1,
    property2: value2,
    …
}
```

Objects

- See **fraction1.js**, **fraction1client.js**

```
$ node fraction1client.js
Numerator 1: 1
Denominator 1: 2
Numerator 2: 3
Denominator 2: 4
f1: 1/2
f2: 3/4
f1 is not identical to f2
f1 is less than f2
-f1: -1/2
f1 + f2: 5/4
f1 - f2: -1/4
f1 * f2: 3/8
f1 / f2: 2/3
$
```

Objects

- **Problem**
  - Instead of calling functions:
    - `f3 = fraction.add(f1, f2);`
  - We want to send messages:
    - `f3 = f1.add(f2);`
- **Solution**
  - The value of an object property can be a function definition…

Objects

- See **fraction2.js**, **fraction2client.js**

```
$ node fraction2client.js
Numerator 1: 1
Denominator 1: 2
Numerator 2: 3
Denominator 2: 4
f1: 1/2
f2: 3/4
f1 is not identical to f2
f1 is less than f2
-f1: -1/2
f1 + f2: 5/4
f1 - f2: -1/4
f1 * f2: 3/8
f1 / f2: 2/3
$
```

Objects

- **Problem**:
  - Space inefficiency…

Objects

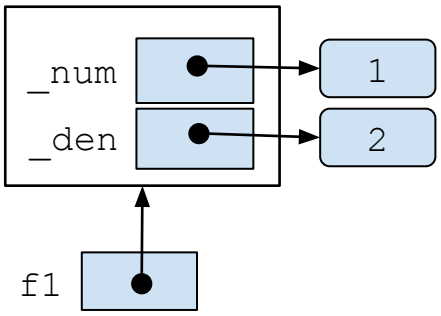
In Python

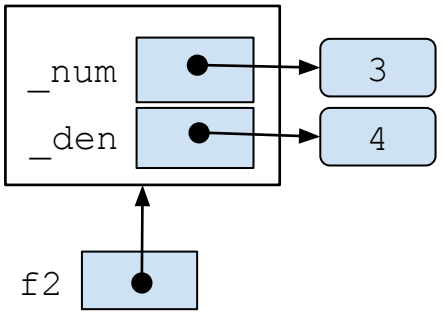
```
f1 = Fraction(1, 2)
f2 = Fraction(3, 4)
```

```
add(self, other):
    …
```

```
sub(self, other):
    …
```

…





Explicit `self` parameter allows `Fraction` objects to share same function defs

Objects

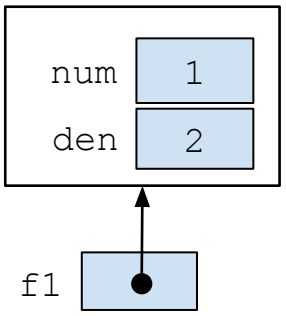
```
Fraction f1 = new Fraction(1, 2);
Fraction f2 = new Fraction(3, 4);
```

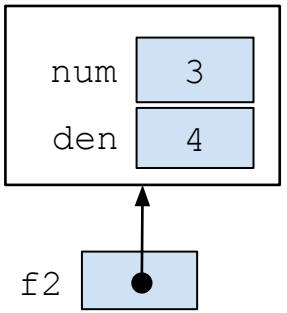
In Java

```
add(this, other)
{…}
```

```
sub(this, other)
{…}
```

…



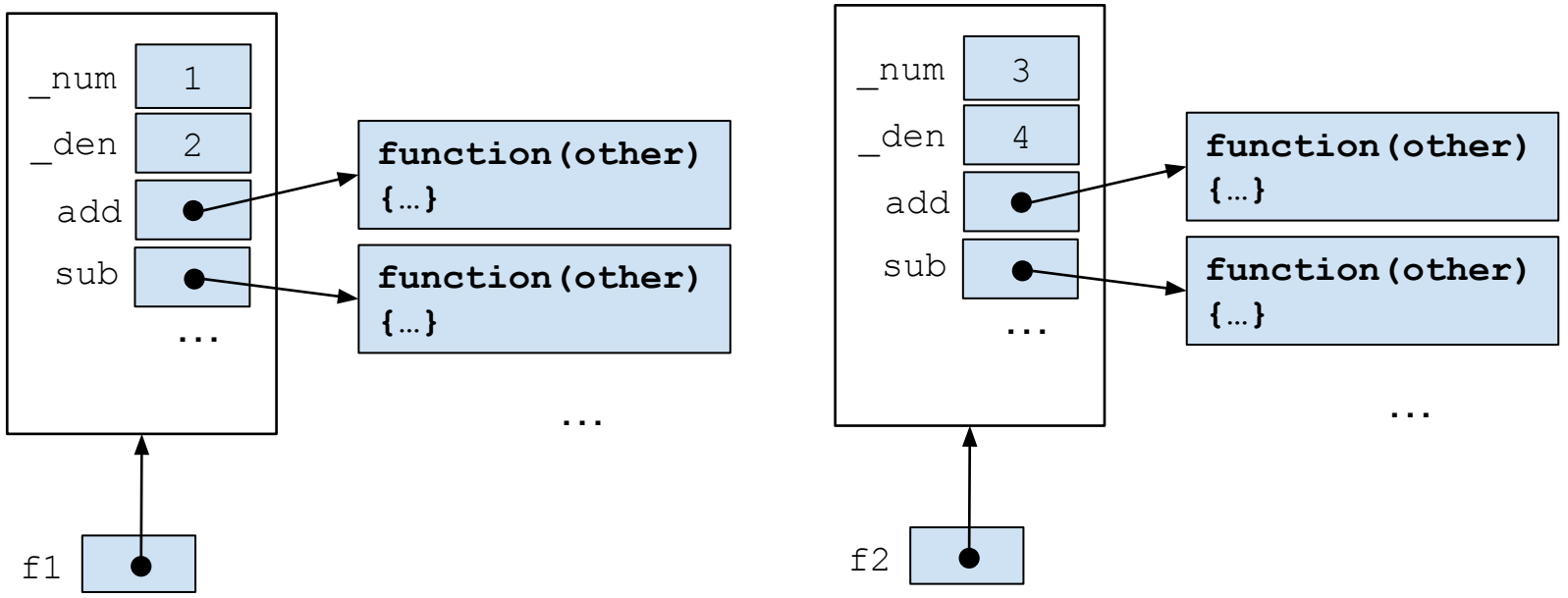


Implicit `this` parameter allows `Fraction` objects to share same method defs

Objects

In JavaScript (so far)

```
let f1 = createFraction(1, 2);
let f2 = createFraction(3, 4);
```



Summary

- We have covered:
  - Modules
  - Objects