JAVASCRIPT*

Интересная задачка для интервью, карринг и частичное применение функции

ИЗ ПЕСОЧНИЦЫ

nmakarov 16 июня 2014 в 10:33

Хожу по job interview. Где-то скучно, где-то весело. Где-то интересно. На одном из таких меня попросили написать функцию, которая умеет складывать два числа. Я написал:

```
it ('should add two numbers', function () {
  var add = function (a,b) {
    return a + b;
  };

  assert.equal(add(2,3), 5);
});
```

А если, говорят, сигнатура функции должна быть типа такой: add(num1)(num2)? Не вопрос, говорю. Думая, что хитрый буржуин хочет проверить, знаю ли я про то, что можно возвращать функции из функций, пишу вот такое:

```
it ('should be called like add(num1)(num2)', function
() {
  var add = function (a) {
```

```
    return function (b) {
      return a + b;
    };
  };

  assert.equal(add(2)(3), 5);
});
```

А вдруг нам первое слагаемое известно заранее, а вот второе будет известно потом, что делать? Ага, думаю, про currying разговор ведут. Вот:

```
  var add3 = add(3);
  assert.equal(add3(4), 7);
  assert.equal(add3(5), 8);
```

Тут вдруг в комнату набежало еще двое, и начали они вчетвером меня распрашивать, махают руками, говорят громко. Не дают сосредоточиться, хотят посмотреть на то, как я думаю. Началось самое интересное.

Спрашивают — а вдруг нужно сложить три числа? Или четыре? Говорю, что надо тогда запоминать состояние, примерно так:

```
  it ('should take random number of digits', function ()
{
    var add = function (a) {
      var sum = a;
      var inner = function (b) {
        if (b) {
          sum += b;
          return inner;
        } else {
          return sum;
        }
      };
      return inner;
    };

    assert.equal(add(2)(3)(), 5);
    assert.equal(add(2)(3)(6)(), 11);
  });
```

Зачем, спрашивают, у тебя внутри if есть? А чтоб внутренняя функция знала, как она вызывается — в цепочке или в самом конце и, соответственно, возвращала бы себя или число. Ладно, говорят, пока ладно. А если опять надо частичное применение? Пишу:

```
    var add2 = add(2);
    assert.equal(add2(6)(), 8);
```

А можно, спрашивают, как-нибудь избавиться от пары пустых скобок вконце? Задумался… Это же функция должна как-то сообразить, в каком контексте ее вызывают… А, есть же волшебный `.valueOf`! И от лишнего if заодно можно избавиться. Ну-ка:

```
var add = function (a) {
  var sum = a;

  var inner = function (b) {
    sum += b;
    return inner;
  };

  inner.valueOf = function () {
    return sum;
  };

  return inner;
};

assert.equal(add(3)(4), 7);
assert.equal(add(3)(5), 8);
assert.equal(add(9)(-5), 4);
assert.equal(add(1)(2)(3), 6);
```

а теперь примени-ка эту add2 к другому числу, скажем, 10 — и чтоб 2+10=12 получилось. Добавляю строку, получаю:

```
    var add2 = add(2);
    assert.equal(add2(6)(), 8);
    assert.equal(add2(10)(), 12);
```

Не работает! Возвращает 18. Это, объясняю, так и задумано — оно внутри запоминает результат последнего сложения и использует для последующих операций. Они — надо исправить, чтоб так явно не запоминало. Хорошо, говорю. Хотите чистых функций? Хотите совсем интересно? Нате цепочку conditional identities:

```
    var add = function (orig) {
      var inner = function (val) {
        return add(parseInt(val+'', 10) == val ?
inner.captured+val : inner.captured);
      };
      inner.captured = orig;
      inner.valueOf = function () {return
inner.captured;};

      return inner;
    };

    assert.equal(add(3)(4), 7);
    assert.equal(add(3)(4)('aa')(5)(), 12);
```

```
    var three = add(3);
    var four = add(4);
    assert.equal(three, 3);
    assert.equal(four, 4);
    assert.equal(three(5), 8);
    assert.equal(three(6), 9);
    assert.equal(three(four), 7);
    assert.equal(three(four)(three(four)), 14);
```

А зачем, спрашивают, нужна вот эта пустая строка:

```
  ... parseInt(val+'', 10) ...
```

Это для принудительного запуска `.valueOf`. Потому что, говорю, если `val` — это функция (что верно для случая, скажем, `three(four)`), то `parseInt` не станет запускать механизм преобразования типов, который в конце концов вызовет `.valueOf`. А `parseInt(func)` — всегда `NaN`.

Смотрю на них — молчат. Не заметили лишнего присваивания, значит. Ладно, надо довести оптимизацию до логического конца. Пишу последний вариант:

```
    var add = function (orig) {
      var inner = function (val) {
        return add(parseInt(val+'', 10) == val ?
orig+val : orig);
      };
```

```
    inner.valueOf = function () {return orig;};

    return inner;
  };
```

Симпатично и минималистично. Тесты в точности те же самые.

Вообще все четырехчасовое интервью получилось весьма полезным. Но окончилось не очень — говорят, тебе у нас не интересно будет. Нам нужны люди, которые будут сидеть с утра до вечера и делать, что сказано, не выпендриваясь и не креативя. Креативом у нас начальство занимается, и у нас его вон уже сколько, новых не надо. Так что тебе вскорости станет скучно, будешь искать себе новую работу. А нам, говорят, текучка ни к чему. А что же тогда, говорю, на интервью позвали, вопросы интересные задавали, задачки решали? А, говорят, звал тебя отдел кадров, а мы думали тебя завалить и тогда тебе не так обидно было бы — не взяли потому, что глупый. А сейчас вот выходит, что не взяли потому, что умный. Легче тебе от этого, спрашивают?

И поехал я домой…

Полный исходник в виде теста на гитхабе: github.com/nmakarov/excercises

Поделиться:

Сохранить:

Комментарии (74)

Похожие публикации

Какие вопросы задавать на собеседовании

40

Mehdzor • 21 февраля 2017 в 10:59

JavaScript. Вопросы на собеседовании

158

AvrGavr • 4 октября 2014 в 01:13

Вопросы на собеседование middle/senior iOS Developer

57

IgorFedorchuk • 23 октября 2013 в 13:42

Популярное за сутки

Наташа — библиотека для извлечения структурированной информации из текстов на русском языке

14

alexkuku • вчера в 16:12

Unit-тестирование скриншотами: преодолеваем звуковой барьер. Расшифровка доклада

4

lahmatiy • вчера в 13:05

Люди не хотят чего-то действительно нового — они хотят привычное, но сделанное иначе

25

ПЕРЕВОД

Smileek • вчера в 10:32

Руководство по SEO JavaScript-сайтов. Часть 2. Проблемы, эксперименты и рекомендации

2

ПЕРЕВОД

ru_vds • вчера в 12:04

Как адаптировать игру на Unity под iPhone X к апрелю

0

P1CACHU • вчера в 16:13

Лучшее на Geektimes

Стивен Хокинг, автор «Краткой истории времени», умер на 77 году жизни

33

HostingManager • вчера в 13:49

Обзор рынка моноколес 2018

70

lozga • вчера в 06:58

«Битва за Telegram»: 35 пользователей подали в суд на ФСБ

40

alizar • вчера в 15:14

Стивен Хокинг и его работа — что дал ученый человечеству?

8

marks • вчера в 14:46

Sunlike — светодиодный свет нового поколения

17

AlexeyNadezhin • вчера в 20:32