

Fizz Buzz

# Table of Contents

1. 仕様	1
2. 設計	1
2.1. TODOリスト	1
2.2. ユースケース図	1
2.3. クラス図	1
2.4. シーケンス図	1
3. 実装	2
3.1. テストコード	2
3.2. プロダクトコード	3
4. 参照	4

# 1. 仕様

- 3で割り切れる場合は「Fizz」を出力する。
- 5で割り切れる場合は「Buzz」を出力する。
- 両方で割り切れる場合は「FizzBuzz」を出力する。
- 指定された回数だけ繰り返し実行する。

## 2. 設計

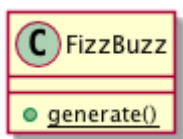
### 2.1. TODOリスト

- ☒ 「Fizz」を出力できるようにする
- ☒ 「Buzz」を出力できるようにする
- ☐ 「FizzBuzz」を出力できるようにする
- ☐ 繰り返し実行できるようにする

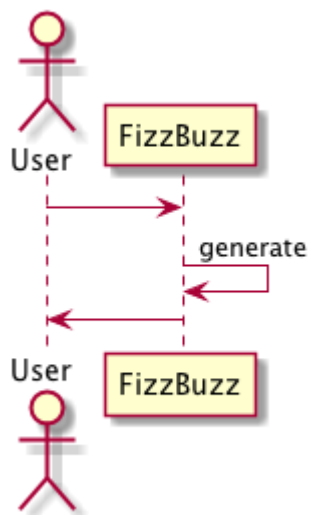
### 2.2. ユースケース図



### 2.3. クラス図



### 2.4. シーケンス図



## 3. 実装

### 3.1. テストコード

```
const chai = require('chai');
const FizzBuzz = require('../src/fizz_buzz');

const { expect } = chai;

describe('Tests FizzBuzz', () => {
  it('3ならばFizzを返す', () => {
    expect(FizzBuzz.generate(3)).to.equal('Fizz');
  });

  it('6ならばFizzを返す', () => {
    expect(FizzBuzz.generate(6)).to.equal('Fizz');
  });

  it('30ならばFizzを返す', () => {
    expect(FizzBuzz.generate(30)).to.equal('Fizz');
  });

  it('5ならばBuzzを返す', () => {
    expect(FizzBuzz.generate(5)).to.equal('Buzz');
  });

  it('10ならばBuzzを返す', () => {
    expect(FizzBuzz.generate(10)).to.equal('Buzz');
  });

  it('50ならばBuzzを返す', () => {
    expect(FizzBuzz.generate(50)).to.equal('Buzz');
  });

  it('15ならばFizzBuzzを返す', () => {
    expect(FizzBuzz.generate(15)).to.equal('FizzBuzz');
  });
});
```

## 3.2. プロダクトコード

```
module.exports = class FizzBuzz {
  static generate(number) {
    let value = number;
    if (number % 3 === 0) {
      value = 'Fizz';
    } else if (number % 5 === 0) {
      value = 'Buzz';
    } else if (number % 3 === 0 && number % 5 === 0) {
      value = 'FizzBuzz';
    }
    return value;
  }
};
```

## 4. 参照

- AsciiDoctor[<http://asciidoctor.org/>]
- PlantUML[<http://www.plantuml.com>]