`this` is tricky in JS

function is an object which can be evoked

```
> var a = [
      function () { console.log('foo'); },
      function () { console.log('bar'); }
  ];
```

```
> a[0]()
  foo
```

```
> a[1].call()
  bar
```

```
> a[0].apply()
  foo
```

```
> a.forEach(function (e) { e(); });
  foo bar
```

`this` is defined at runtime ...

```
> var printThis = function () { console.log(this); };


> printThis()
  global


> var obj = { printMe: printThis };
> obj.printMe()
  { printMe: [Function] }
```

… and this can be overwritten

```
1:  function once (fn) {
2:    var executed = false;
3:
4:    return function () {
5:      return executed ? undefined :
6:        ((executed = true), fn.apply(this, arguments));
7:    };
8:  }
```

```
> var doOnce = once(function () { console.log(this); });

> doOnce(); doOnce();
  global
  undefined
```