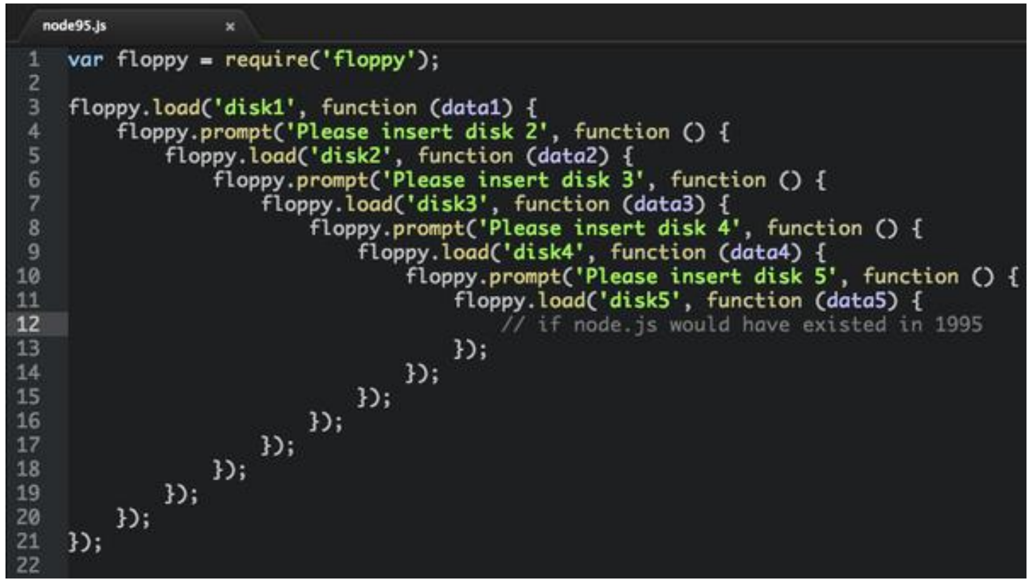
Promise

YAPP

박 영 준

Promise란?

비동기에서의 Callback function


node95.js

```
var floppy = require('floppy');

floppy.load('disk1', function (data1) {
    floppy.prompt('Please insert disk 2', function () {
        floppy.load('disk2', function (data2) {
            floppy.prompt('Please insert disk 3', function () {
                floppy.load('disk3', function (data3) {
                    floppy.prompt('Please insert disk 4', function () {
                        floppy.load('disk4', function (data4) {
                            floppy.prompt('Please insert disk 5', function () {
                                floppy.load('disk5', function (data5) {
                                    // if node.js would have existed in 1995
                                });
                            });
                        });
                    });
                });
            });
        });
    });
});
```

```
$.ajax({
  url : '/posts',
  type: 'GET',
  success : function(result1){
    var posts = result1.data;
    $.ajax({
      url : '/posts/'+posts[0]._id,
      type: 'GET',
      success : function(result2){
        var post = result2.data;
        $.ajax({
          url : '/users/'+post.author._id,
          type: 'GET',
          success : function(result3){
           var user = result3.data;
           // user로 할일 - 3
          }
        });
        // 그외 post로 할일 - 2
      }
    });
    // 그외 posts로 할일 - 1
  }
});
```

가독성..

Solutions

Inline 함수 사용

모듈화

코드의 간결화

Promise 사용

Promise란?

ES6 전

jQuery에서의 Deffered

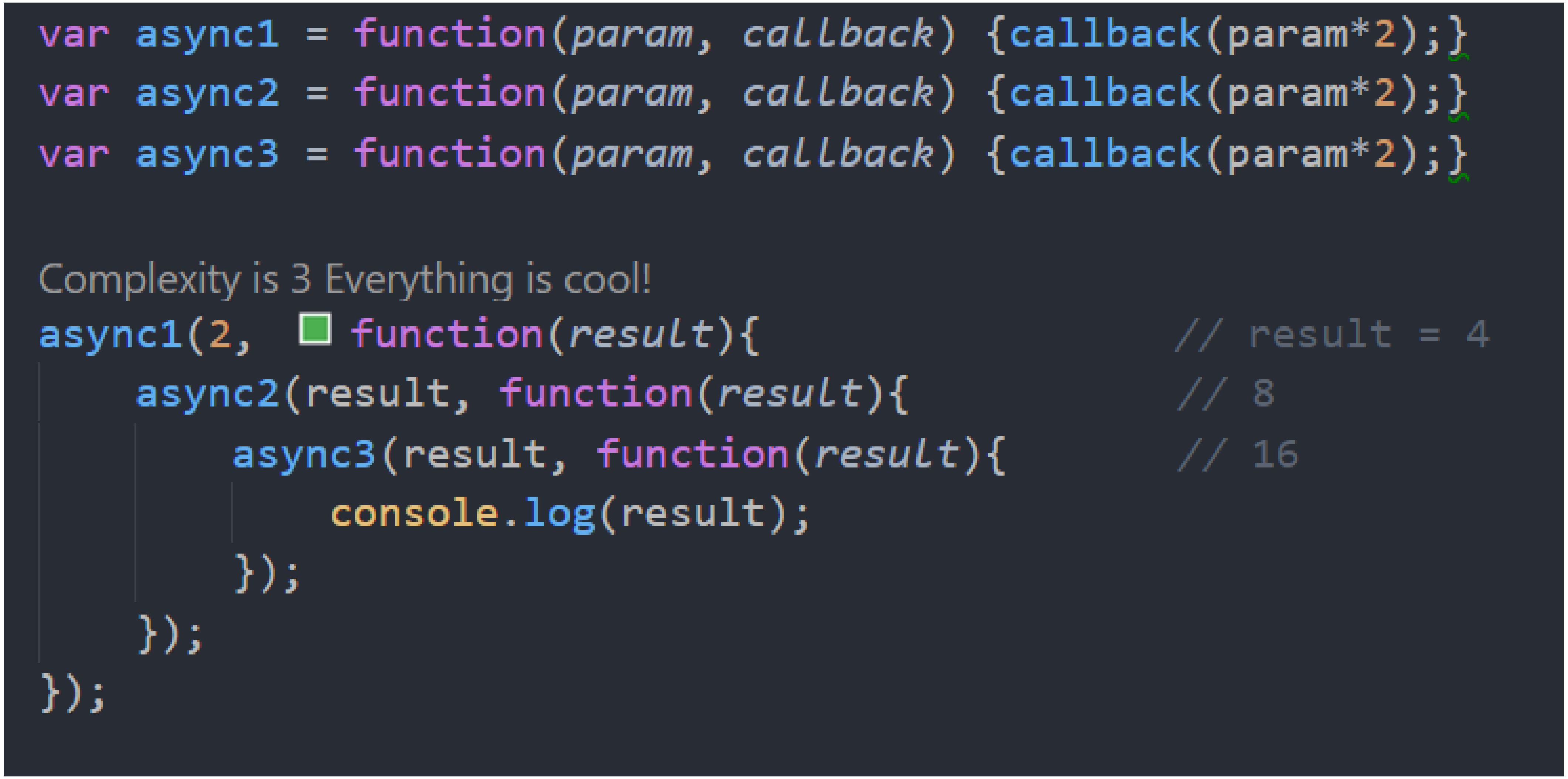
Javascript Q, Vow, Bluebird 라이브러리

ES6 후

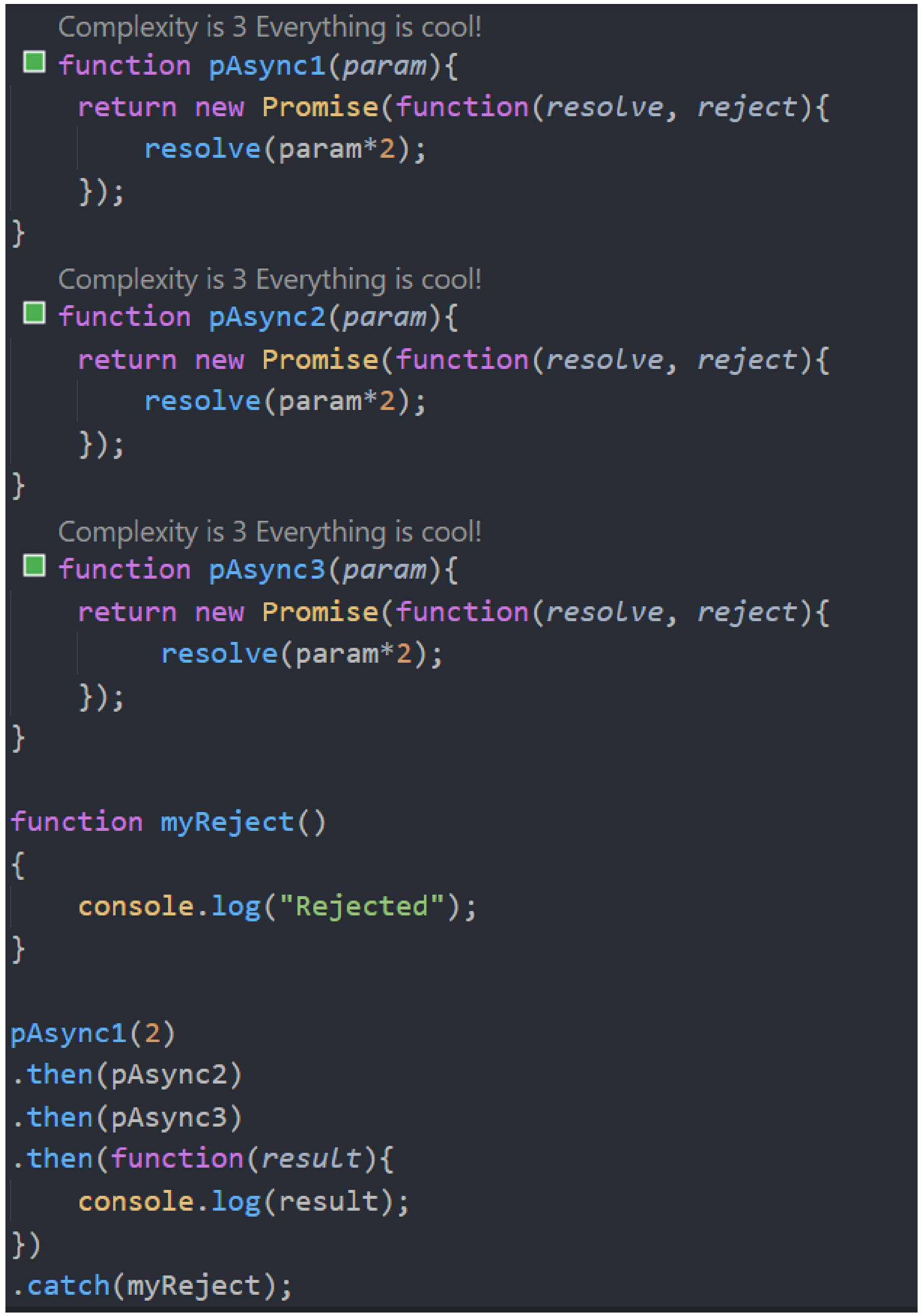
크롬 정식 지원

Node.js 지원

Sample

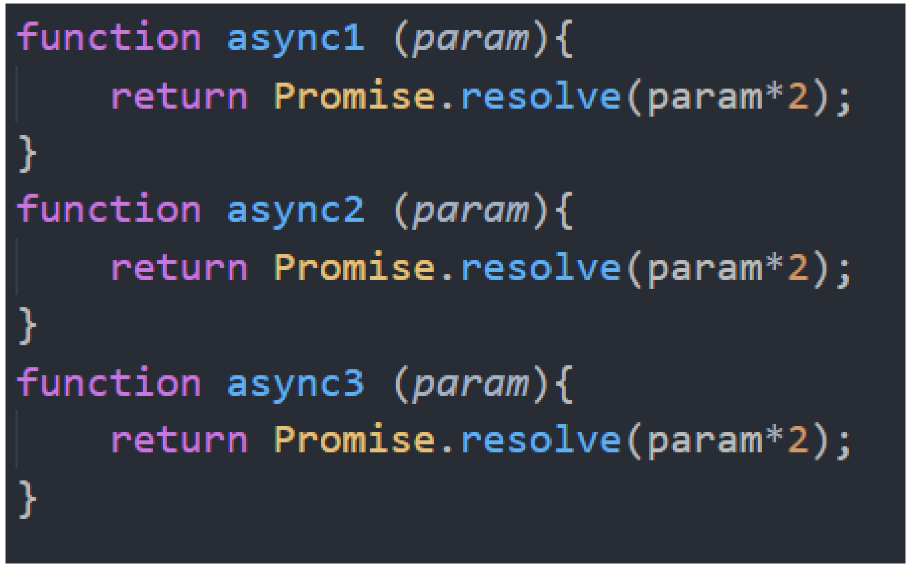


```
var async1 = function(param, callback) {callback(param*2);}
var async2 = function(param, callback) {callback(param*2);}
var async3 = function(param, callback) {callback(param*2);}

Complexity is 3 Everything is cool!
async1(2, 🟩 function(result){                    // result = 4
    async2(result, function(result){              // 8
        async3(result, function(result){          // 16
            console.log(result);
        });
    });
});
```

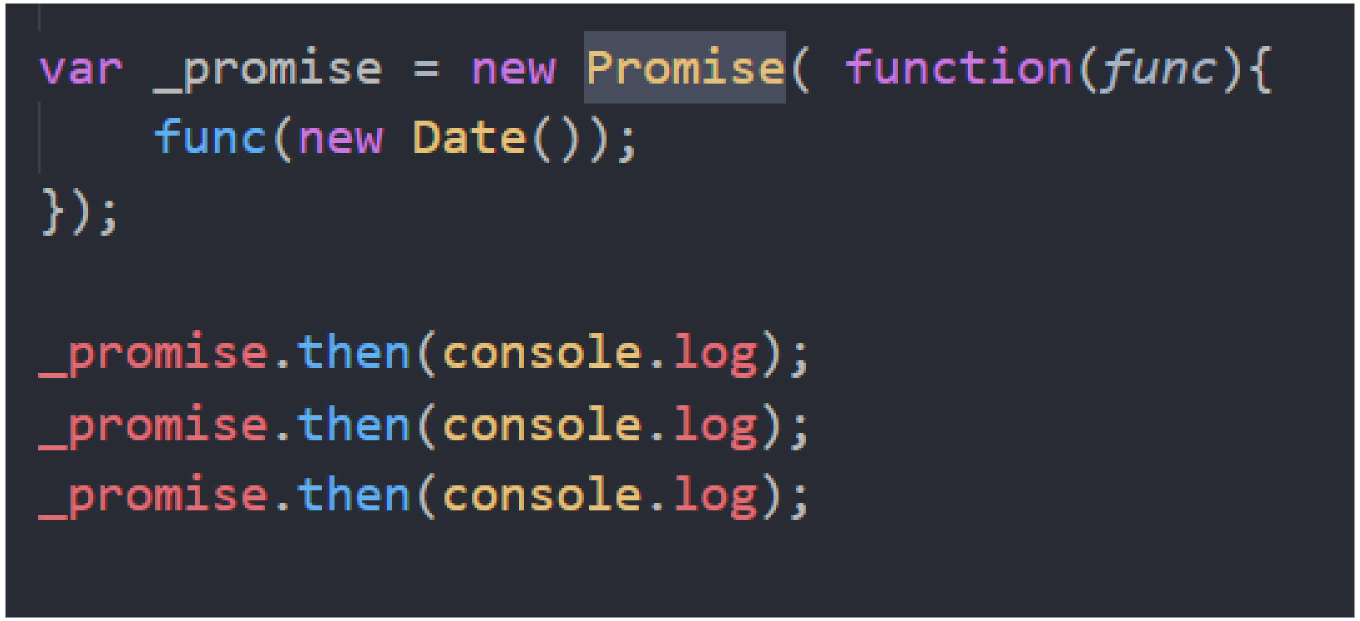


```
   Complexity is 3 Everything is cool!
function pAsync1(param){
    return new Promise(function(resolve, reject){
        resolve(param*2);
    });
}
   Complexity is 3 Everything is cool!
function pAsync2(param){
    return new Promise(function(resolve, reject){
        resolve(param*2);
    });
}
   Complexity is 3 Everything is cool!
function pAsync3(param){
    return new Promise(function(resolve, reject){
         resolve(param*2);
    });
}

function myReject()
{
    console.log("Rejected");
}

pAsync1(2)
.then(pAsync2)
.then(pAsync3)
.then(function(result){
    console.log(result);
})
.catch(myReject);
```

Promise sample

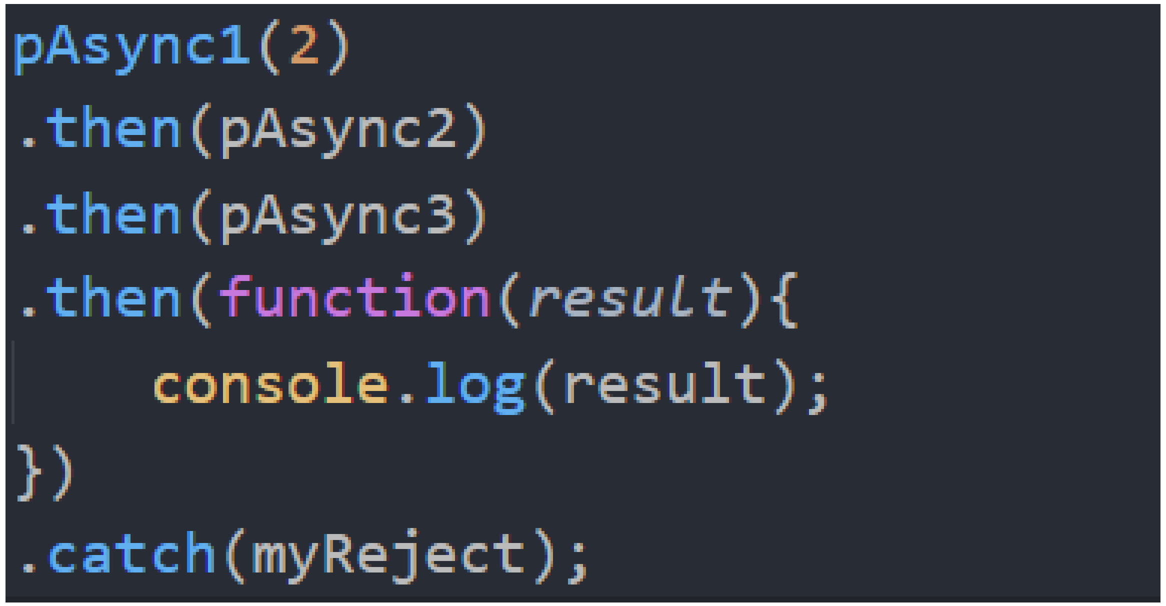


```
function async1 (param){
    return Promise.resolve(param*2);
}
function async2 (param){
    return Promise.resolve(param*2);
}
function async3 (param){
    return Promise.resolve(param*2);
}
```

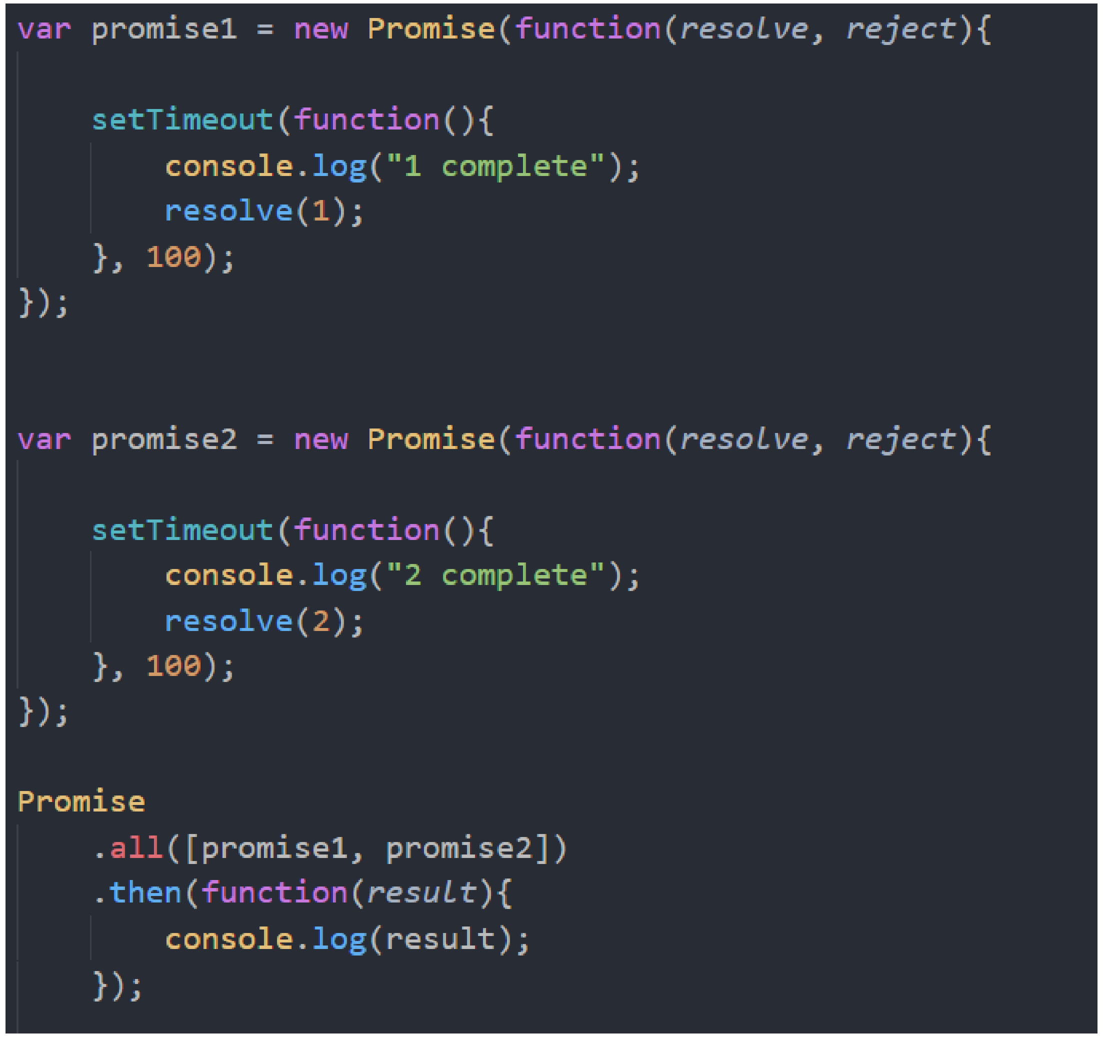


```
var _promise = new Promise( function(func){
    func(new Date());
});

_promise.then(console.log);
_promise.then(console.log);
_promise.then(console.log);
```

Promise sample

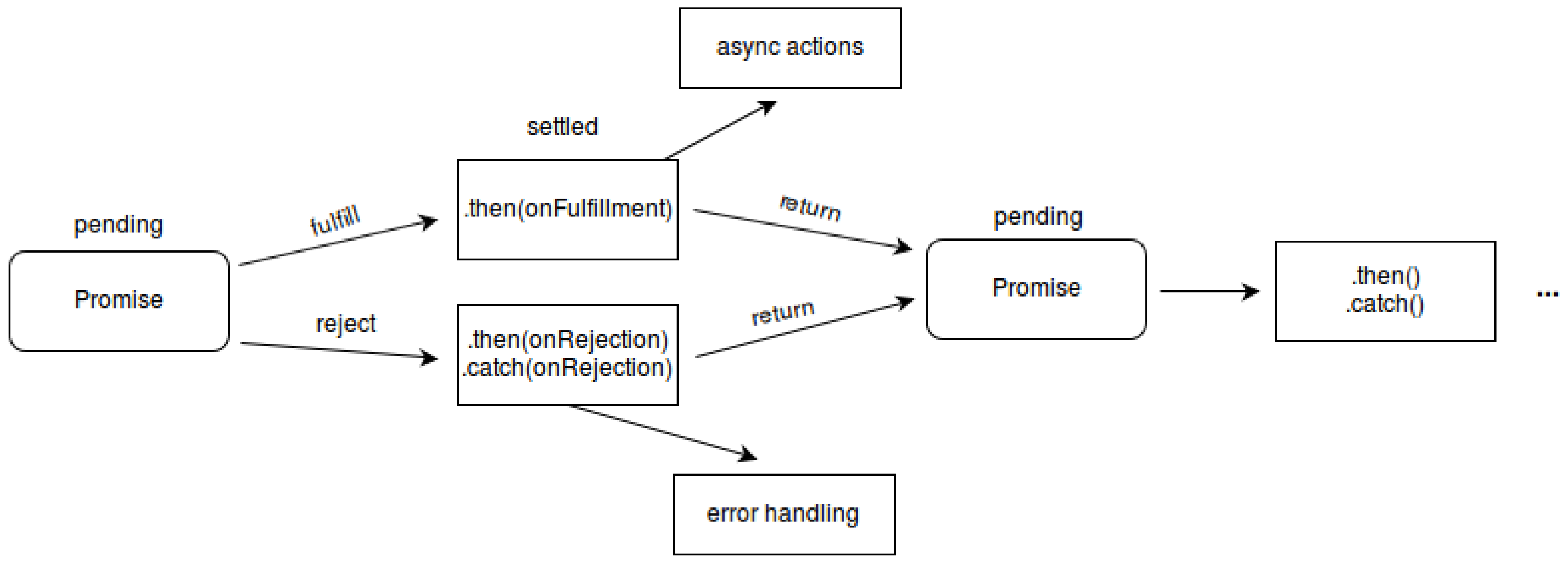


```
pAsync1(2)
.then(pAsync2)
.then(pAsync3)
.then(function(result){
    console.log(result);
})
.catch(myReject);
```



```
var promise1 = new Promise(function(resolve, reject){

    setTimeout(function(){
        console.log("1 complete");
        resolve(1);
    }, 100);
});


var promise2 = new Promise(function(resolve, reject){

    setTimeout(function(){
        console.log("2 complete");
        resolve(2);
    }, 100);
});

Promise
    .all([promise1, promise2])
    .then(function(result){
        console.log(result);
    });
```

Promise의 상태



async actions
settled
.then(onFulfillment)
pending
Promise
fulfill
reject
return
return
.then(onRejection)
.catch(onRejection)
pending
Promise
.then()
.catch()
...
error handling

감사합니다