




**Q** - A library for promises

Anil BALANI  
Julian PARMENTIER-BERLIN  
Jaymin PATEL  
Vikalp PATEL

Q

The “pyramid of doom”

A yellow pyramid shape composed of several vertical bars of decreasing height from left to right, representing the stack of recursive calls.

```
step1(function (value1) {  
  step2(value1, function(value2) {  
    step3(value2, function(value3) {  
      step4(value3, function(value4) {  
        // Do something with value4  
      });  
    });  
  });  
});
```

## Flatten the pyramid

```
Q.fcall(promisedStep1)
  .then(promisedStep2)
  .then(promisedStep3)
  .then(promisedStep4)
  .then(function (value4) {
    // Do something with value4
  })
  .catch(function (error) {
    // Handle any error from all above steps
  })
  .done();
```

## The competitors - Async



### Asynchronous

Async provides the developer with asynchronous versions of control structures and aggregate operations.

```
async.waterfall([
  function(callback) {
    callback(null, 'one', 'two');
  },
  function(arg1, arg2, callback) {
    // arg1 now equals 'one' and arg2 now
    // equals 'two'
    callback(null, 'done');
  },
  function (err, result) {
    // result now equals 'done'
  });
]);
```



### Callbacks

Async isn't using promises, you are still working with callbacks.

## The competitors - Bluebird



### Performance oriented

- 100 times faster than Q (according to Bluebird).
- Much more debuggable with a full stack trace



### Cons

- Security flaws

## The competitors - Step



### Pros

- Easy to use, no learning required.
- Pretty much copy-and-paste if converting an existing project

```
Step(  
  function readSelf() {  
    fs.readFile(__filename, this);  
  },  
  function capitalize(err, text) {  
    if (err) throw err;  
    return text.toUpperCase();  
  },  
  function showIt(err, newText)  
{  
  if (err) throw err;  
  console.log(newText);  
}  
);
```



### Cons

- No common error handler
- Harder to indent the step function properly

# Ressources

That's a lot of **links**!



## Q - Npm

<https://www.npmjs.com/package/q>  
Official npm page



## Q - Wiki

<https://github.com/kriskowal/q/wiki>  
Official wiki page with examples, libraries etc..



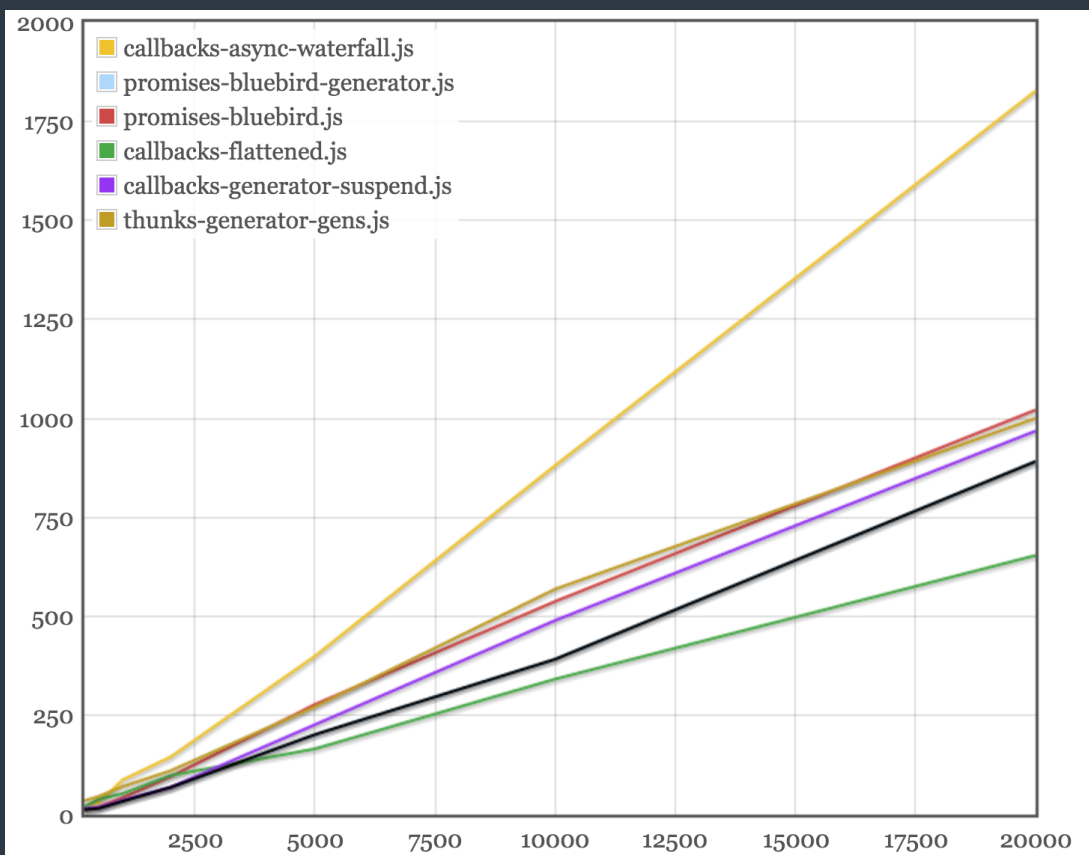
## Q - Documentation

<https://documentup.com/kriskowal/q>  
Alternative link for documentation

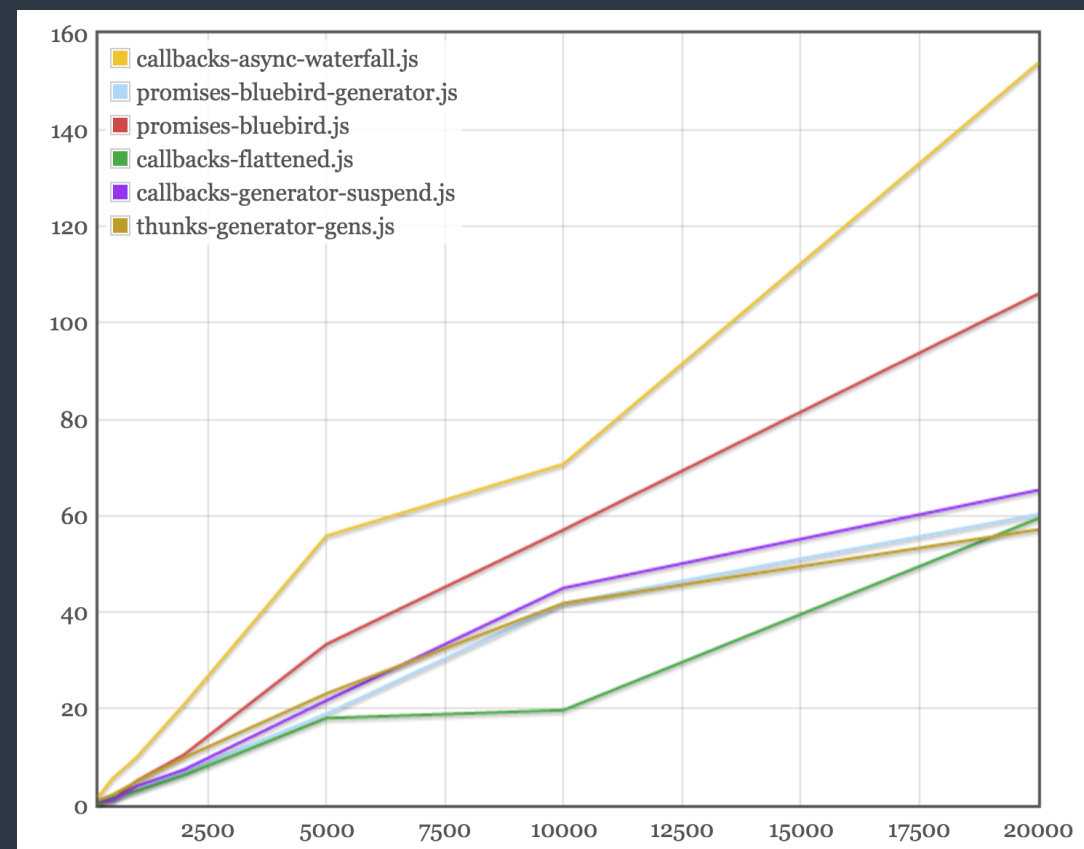
# Promises - Better and faster

Because **speed** is key

Time (ms)



Memory (MB)





# Quick note

If you want to know more about **Promises** (we swear, they're cool!)  
please visit : <https://promisesaplus.com>



# Any questions?

WHAT

WHY

WHERE

HOW