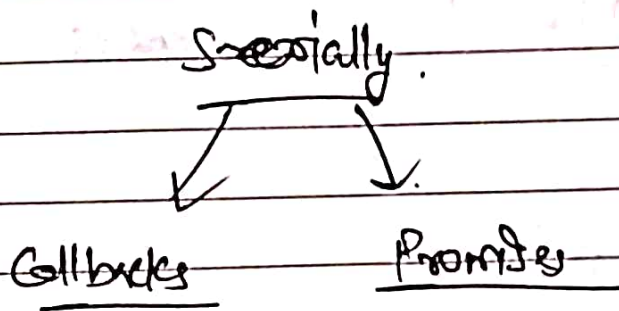


Reading File Serially

(i.e. reading orderly, earlier we are getting files randomly, but now f1, f2, f3).



Interview How to execute serial function & how to execute a Parallel function.

Thru Callbacks

```
cb1()
{
  cb2()
  }
cb2()
{
  cb3()
  }
cb3()
{
  console.log(data);
}
```

calling Async fn. (callbacks) serially.

Date

```
fs.readFile ('f1.txt', cb1);
```

```
function cb1 (err, data)
```

```
{  
  if (err)  
  {  
    c.log(err);  
  }  
}
```

```
else  
{  
  c.log(" " + data);  
}
```

```
fs.readFile ('f2.txt', cb2);  
}
```

```
}
```

```
function cb2 (err, data)
```

```
{  
  if (err)  
  {  
    c.log(err);  
  }  
}
```

```
else  
{  
  c.log(" " + data);  
}
```

```
fs.readFile ('f3.txt', cb3);  
}
```

```
function cb3 (err, data)
```

```
{  
  if (err)  
  {  
    c.log(err);  
  }  
}
```

```
else  
{  
  c.log(" " + data);  
}
```

```
fs.readFile ('f4.txt', cb4);  
}
```

```
}
```


Serially

Date

Thou Promises

```
let fs = require('fs')
```

```
c.log('Before');
```

```
let f1p = fs.promises.readFile('f1.txt'); Pending
```

```
f1p.then(cb1); resolve
```

```
function cb1(data) {
```

```
  c.log("File Data → " + data);
```

```
  let f2p = fs.promises.readFile('f2.txt'); Pending
```

```
  f2p.then(cb2); resolve
```

```
function cb2(data) {
```

```
  c.log("File Data → " + data);
```

```
  let f3p = fs.promises.readFile('f3.txt'); Pending
```

```
  f3p.then(cb3); resolve
```

```
function cb3(data) {
```

```
  c.log("File Data → " + data);
```

```
}
```

```
c.log('After');
```

Qp-

Before

After

File Data → this is file 1

File Data → this is file 2

File Data → this is file 3

→ This orders with never change, ∴ we are running it Serially.

Syntactic sugar

Date

```
fp.then(cb).then(cb2).then(cb3).catch(function(err) {  
    c.log(err);  
})
```

(o/r) same, we just wrote it in a different way).