JÖNKÖPING UNIVERSITY

School of Engineering

DEPENDENCY INJECTION IN NODE.JS

Peter Larsson-Green

Jönköping University

Spring 2019

WHAT IS DEPENDENCY INJECTION?

A way to avoid hard-coding dependencies.

- Very similar to how we generalize functions.

```
function contains10(theArray){
  for(const value of theArray){
    if(value == 10){
      return true
    }
  }
  return false
}
```

```
function contains(theArray, lookingFor){
  for(const value of theArray){
    if(value == lookingFor){
      return true
    }
  }
  return false
}
```

```
const exists = contains10([4, 7])
```

```
const exists = contains([4, 7], 10)
```

WHAT IS DEPENDENCY INJECTION?

A way to avoid hard-coding dependencies.

- Very similar to how we generalize functions.


```
const accountRepo = require('../dal/account-repository')
const blogRepo = require('../dal/blog-repository')
```

bll/blog-manager.js

bll/album-manager.js

```
const accountRepo = require('../dal/account-repository')
const albumRepo = require('../dal/album-repository')
```

WHAT IS DEPENDENCY INJECTION?

A way to avoid hard-coding dependencies.

- Very similar to how we generalize functions.


```
const accounts = [{id: 1, username: "Bob"}]
exports.getAccountById = function(id, cb){
  cb(accounts.find(a => a.id == id))
}
```

dal/account-repository-memory.js

```
const mysql = require('mysql')
exports.getAccountById = function(id, cb){
  mysql.newConn(...).query(..., cb)
}
```

dal/account-repository-db.js

AWILIX

A Dependency Injection library/container for Node.js.

- `npm install awilix`

```
const blogRepo = require('../dal/blog-repo')
exports.getAllBlogs = function(callback){
  blogRepo.getAll(callback)
}
```

bll/blog-manager.js

bll/blog-manager.js

```
module.exports = function(container){
 return {
  getAllBlogs: function(callback){
   container.blogRepo.getAll(callback)
  }
 }
}
```

AWILIX

```
const awilix = require('awilix')
const blogRepoFun = require('dal/blog-repo')
const blogManagerFun = require('bll/blog-manager')
const container = awilix.createContainer()
container.register('blogRepo', awilix.asFunction(blogRepoFun))
container.register('blogManager', awilix.asFunction(blogManagerFun))
const blogManager = container.resolve('blogManager')
```

main.js

bll/blog-manager.js

```
module.exports = function(container){
 return {
  getAllBlogs: function(callback){
   container.blogRepo.getAll(callback)
  }
 }
}
```

AWILIX

```
const {a, b} = {a: 1, b: 2}
const three = a + b
```

bll/blog-manager.js

```
module.exports = function({blogRepo}){
 return {
  getAllBlogs = function(callback){
   blogRepo.getAll(callback)
  }
 }
}
```

bll/blog-manager.js

```
module.exports = function(container){
 return {
  getAllBlogs: function(callback){
   container.blogRepo.getAll(callback)
  }
 }
}
```