FORMATION FULL STACK

הרווחה, משרד העבודה
והשירותים החברתיים

Node.js

By Charles Cohen

Charles Cohen
Senior Full Stack Developer
charlesc@edandweb.co.il
050-4449764

Programme du cours Node.js

1. Introduction to Node.js
   - What is Node and what is it not ?
   - Node.js Features ?
   - Our first Node.js script: Hello World
   - Building a web server in Node.js
   - Debugging node applications
2. Building your Stack
   - Pulling in other libraries
   - Building custom libraries
   - A-synchronicity and callbacks
   - Blocking vs. non-blocking I/O
   - Working within the event loop
3. Modular JavaScript with Node.js
   - Writing Modular JavaScript with Node.js
   - Core Modules
   - Installing Packages
   - Publishing packages
4. Avoiding common pitfalls with Async.js
   - Introducing the Asynchronous problem
   - Async.js Library to the rescue
   - Collections
   - Flow Controllers
5. Working with the file system
   - Files manipulations
   - Folder manipulations
   - Putting the file-system module together Async.js
6. Building Web applications with the Express Framework
   - Introduction to Express, installation and basic setup
   - Application configuration
   - Routing
   - Views and Templating options
   - Persistence with Cookies, In-Memory Sessions and session-stores
   - Social Authentication with Passport.js
7. Connecting MySQL Server
   - Database connection
   - A-synchronicity Queries from node.js

Cours 7

MongoDB et Node.js…

Plan du module Node.js

Cours	Date	Cours
1	Lun. 20/05	• Introduction to Node.js (1)
2	Mer. 29/05	Building your Stack (2) • Command Line • File System • Arrow Functions • Modular JavaScript with Node.js
3	Mer. 05/06	• Asynchronous JS • Consuming API – HTTP requests
4	Lun. 10/06	• Consuming API – HTTP requests • Exercises – Notes & Weather
5	Lun. 17/06	• Express framework – templates with Handlebars • Building a get API entry point based on previous exercices (get coordinates and weather from mapbox.com and darksky.net) and returning data as JSON • Implement a search address box that consuming the GET API
6	Lun. 01/07	• Promise, await, async, Async module … • Avoiding common pitfalls with Async.js
7	Lun 08/07	• MongoDB ½
8	Lun 15/07	• MongoDB 2/2
9	Lun 22/07	• Working with MongoDB 1/2
10	Lun 15/07	• Working with MongoDB 2/2

Module Node.js & MongoDB

Cours	Date	Cours
7	Lun. 08/07	• Introduction to MongoDB • Installation MongoDB server locally • Installation MongoDB GUI • MongoDB - working with Node.js • CRUD Operations. • Exrcice: Integrate into our Tasks application a DB
8	Lun 15/07	• MongoDB Basics • API mongoose ½
9	Lun 22/07	• API Mongoose 2/2 • Async/Await integration with mongoDB
10	Lun 29/07	• Authentication ?

Environnement de travail

Editeur de code:

- Visual Studio Code
  - Extensions:

Liens Utiles:

- Node.js: https://nodejs.org/
- Moteur V8 Javascript: https://v8.dev/
- Express : http://expressjs.com

Résumé du cours 5

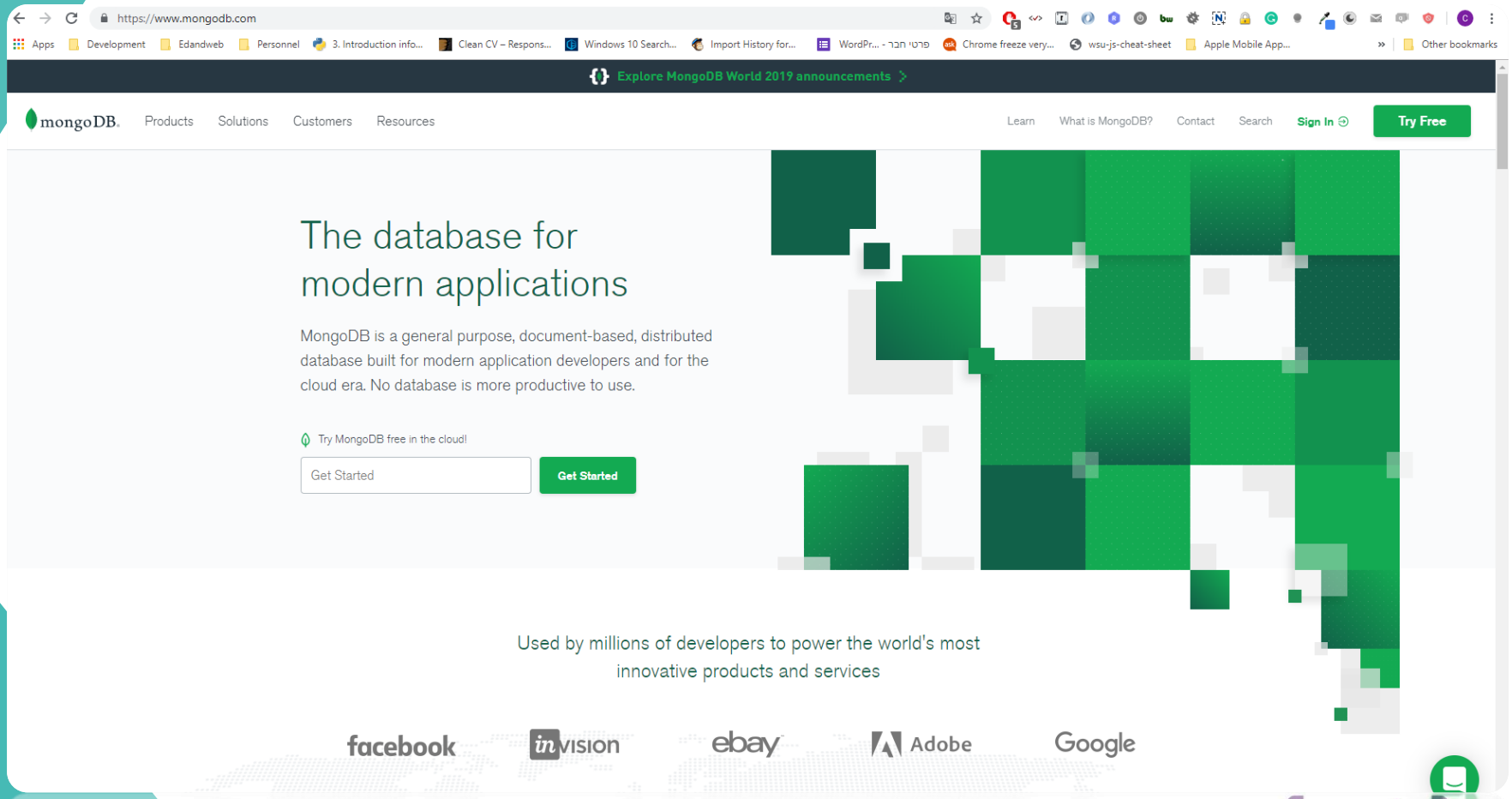
- Retour sur les framework express et sa configuration (voir résumé cours 4)
- Utilisation des blocks pour étendre un template avec Handlebars dans base.hbs (`{{{> @partial-block}}}`) et dans index.hbs (`{{#> base}}Content{{/base}}`)
- Utilisation des boucles dans un template:

```
{{#each peoples}}
<li>Nom: {{this.name}}, Age: {{this.age}}, Country: {{this.country}}</li>
{{/each}}
```

- Création d'un point d'entré d'API GET basé sur les API mapbox.com et darksky.net
- Réalisation d'un formulaire de recherche faisant appel à l'API GET pour retourner la météo en function de l'adresse tappée dans le navigateur

Mongo DB

http://www.mongodb.com



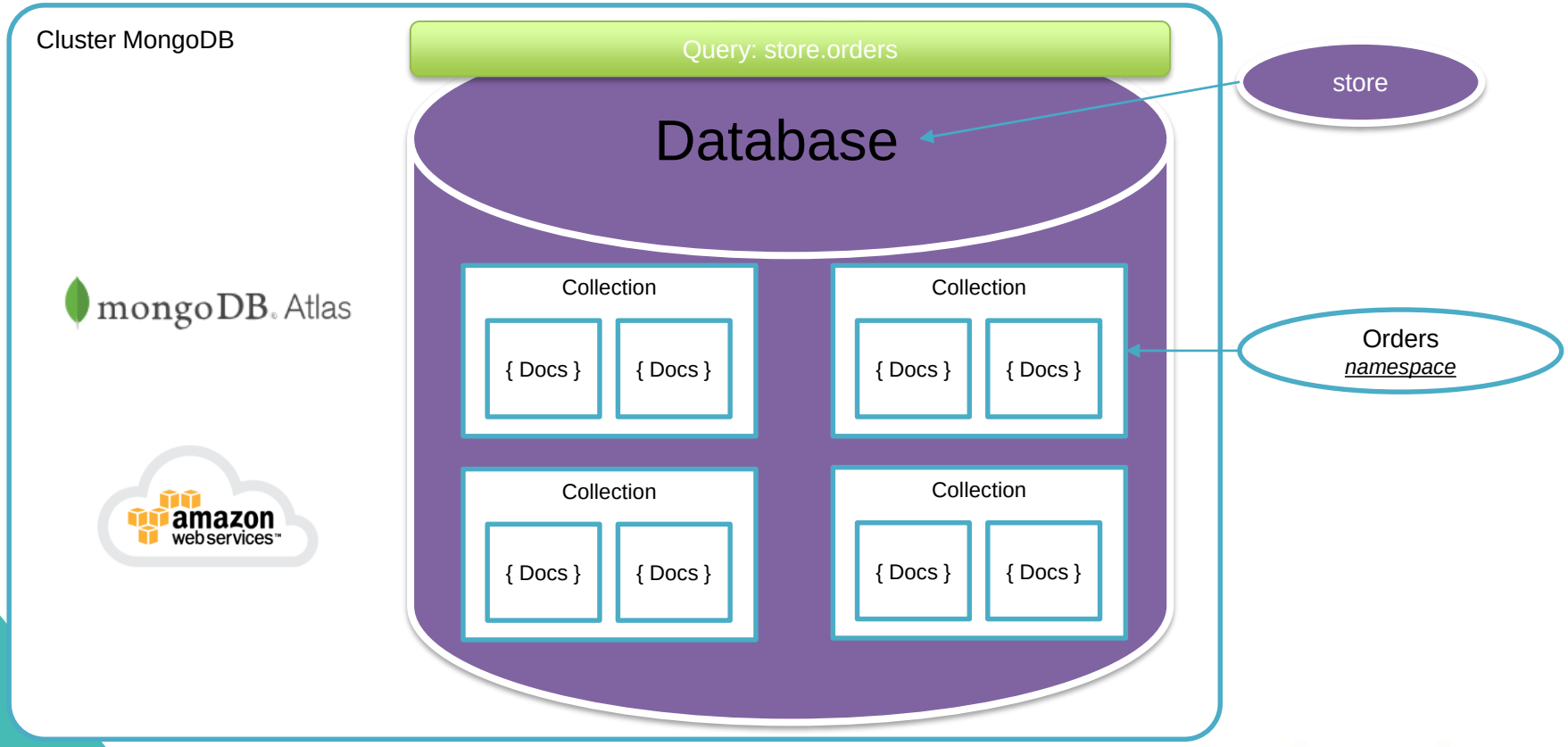
Introduction à MongoDB

- What is Mongo DB ?
- Database, Collections, Documents ?
- Field Types
- Database GUI
- Connection to a cluster
- Schema
- Documents
- Filter

What is MongoDB ? (Wikipedia)

- **MongoDB** is a cross-platform document-oriented database program.
- Classified as a NoSQL database program, MongoDB uses JSON-like documents with schemata.
- MongoDB is developed by MongoDB Inc.and licensed under the Server Side Public License (SSPL).

Database, Collections, Documents



Type de champs

- Scalar
  - Int32
  - Double
  - String
  - Date
- Document – a field can contain a document
- Array
- Coordinates

Database & GUI Installation

FOR DATA VISUALISATION AND ANALYSE, WE'LL USE COMPASS
(NOT COMMUNITY EDITION)

Installation serveur local

https://www.mongodb.com/what-is-mongodb

Apps | Development | Edandweb | Personnel | 3. Introduction info... | Clean CV – Respons... | Windows 10 Search... | Import History for... | WordPr... - פרטי חבר | Chrome freeze very... | wsu-js-cheat-sheet | Apple Mobile App... | Other bookmarks

Explore MongoDB World 2019 announcements

mongoDB. Products Solutions Customers Resources Learn What is MongoDB? Contact Search Sign In Try Free

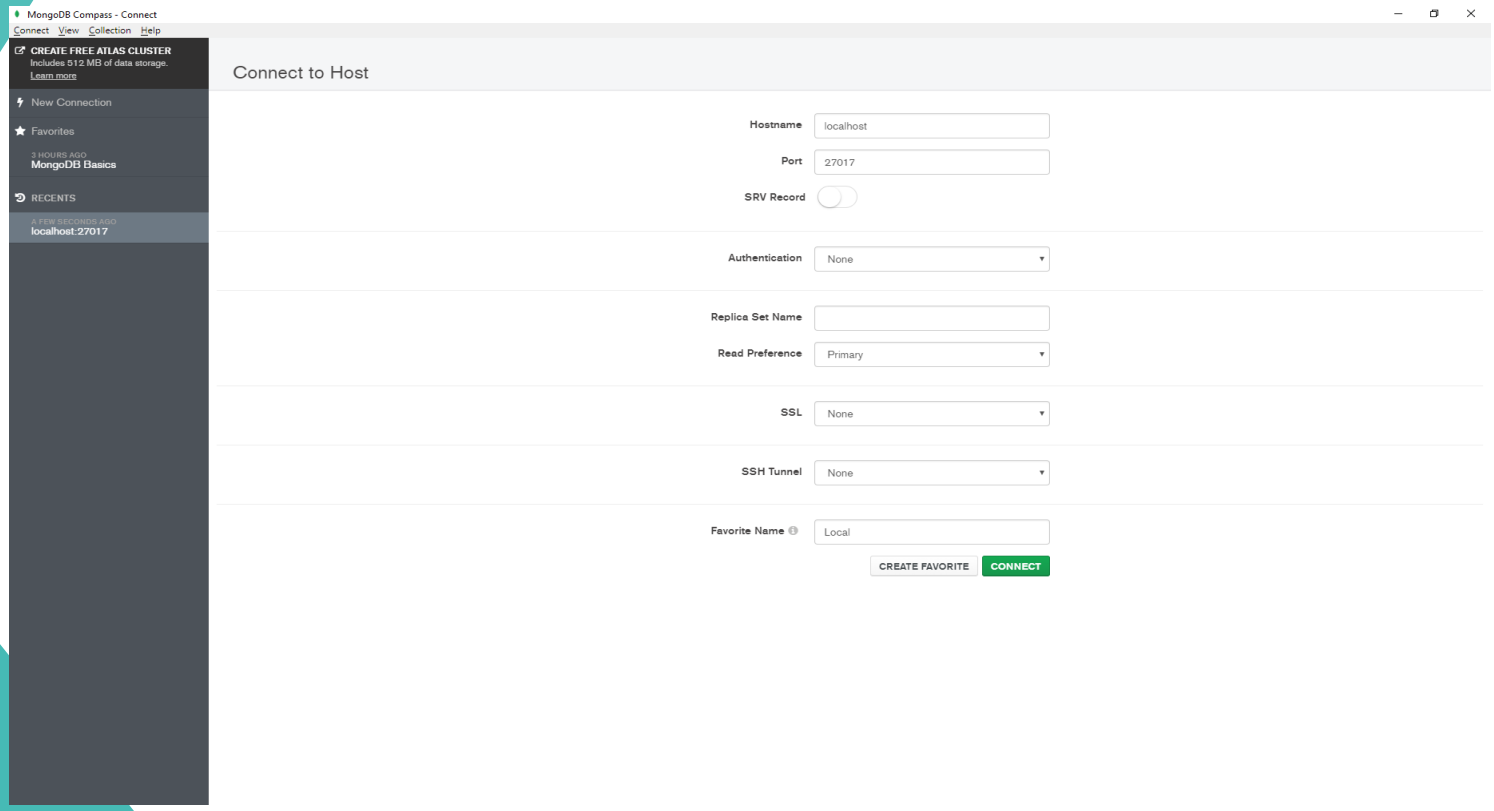
What is MongoDB?

MongoDB is a document database with the scalability and flexibility that you want with the querying and indexing that you need

Download the latest release 4.0 | Explore the documentation

- MongoDB **stores data in flexible, JSON-like documents**, meaning fields can vary from document to document and data structure can be changed over time
- The document model **maps to the objects in your application code**, making data easy to work with
- **Ad hoc queries, indexing, and real time aggregation** provide powerful ways to access and analyze your data
- MongoDB is a **distributed database at its core**, so high availability, horizontal

Cluster - Connection



Schema

- Fields of the collection
- Type of the field
- Proportion of Values for each field (80%: int32, 20%: string)
- Range of values of each field
- Select a specific values (filter)
- Select with the mouse a part of the values to focus on
- Define for Geo Spatial data with the shift key and the mouse (radius selection)

Filter

- {‘’field’’: ‘’search’’}
- {‘’field’’: { ‘’$gte’’: 1965, ‘’lt’’:1970 } }

MongoDB – driver officel Node.js

https://docs.mongodb.com/ecosystem/drivers/node/

Apps | Development | Edandweb | Personnel | 3. Introduction info... | Clean CV – Respons... | Windows 10 Search... | Import History for... | WordPr... - פרטי חבר | Chrome freeze very... | wsu-js-cheat-sheet | Apple Mobile App... | Other bookmarks

mongoDB. | Documentation

SERVER | DRIVERS | CLOUD | TOOLS | GUIDES | Get MongoDB | Search Documentation

Close ×

MongoDB Ecosystem

- MongoDB Drivers and ODM
  - C Driver
  - C++ Driver
  - C# and .NET Driver
  - Go Driver
  - Java Driver
  - Node.js Driver
  - Perl Driver
  - PHP Driver
  - Python Drivers
  - Ruby Driver
  - Mongoid ODM
  - Scala Driver
  - Community Supported Drivers Reference
  - MongoDB Drivers Specs
  - Driver Compatibility
- MongoDB Integration and Tools
- Platforms
- Use Cases

MongoDB Drivers and ODM > MongoDB Node.js Driver

MongoDB Node.js Driver

On this page

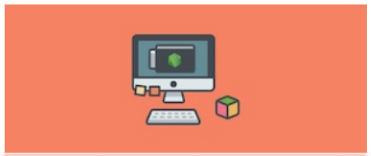
- Introduction
- Take the free online course taught by MongoDB
- Installation
- Connect to MongoDB Atlas
- Compatibility
- How to get help

Introduction

The official MongoDB Node.js driver provides both callback-based and Promise-based interaction with MongoDB, allowing applications to take full advantage of the new features in ES6. The 2.x series of the driver is powered by a brand new core driver and BSON library.

- Tutorials
- Usage Guide
- API Reference
- Changelog
- Source Code

Take the free online course taught by MongoDB



M220JS: MongoDB for JavaScript Developers

Learn the essentials of Node.js application development with MongoDB.

Was this page helpful?

Connecting to MongoDB

```
const mongodb = require('mongodb')
const MongoClient = mongodb.MongoClient

const connectionURL = 'mongodb://127.0.0.1:27017'
const databaseName = 'task-manager'

MongoClient.connect(connectionURL, { useNewUrlParser: true }, (error, client) =>
{
    if (error) {
        return console.log('Unable to connect to database’)
    }

    console.log('Connected correctly !');
});
```

Inserting document into MongoDB

```
const mongodb = require('mongodb')
const MongoClient = mongodb.MongoClient

const connectionURL = 'mongodb://127.0.0.1:27017'
const databaseName = 'task-manager'

MongoClient.connect(connectionURL, { useNewUrlParser: true }, (error, client) => {
    if (error) {
        return console.log('Unable to connect to database’)
    }

    const db = client.db(databaseName)
    db.collection('users').insertOne(
    {
        name: 'Charles’,
        age: 36
    }, (error, result) => {
         if (error) {
              return console.log('Unable to insert user’)
         }

         console.log(result.ops);
    })
});
```

Find document

```
const mongodb = require('mongodb')
const MongoClient = mongodb.MongoClient

const connectionURL = 'mongodb://127.0.0.1:27017'
const databaseName = 'task-manager'

MongoClient.connect(connectionURL, { useNewUrlParser: true }, (error, client) => {
    if (error) {
        return console.log('Unable to connect to database’)
    }

    const db = client.db(databaseName)
    db.collection('users').findOne({name: 'Charles'}, (error, user) => {
        if (error) {
            return console.log('Unable to find the user’)
        }
        console.log(user);
    })

    // Pointer - go to doc
    db.collection('tasks').find({completed: false}).toArray()
});
```

Update a document

```
const mongodb = require('mongodb')
const MongoClient = mongodb.MongoClient

const connectionURL = 'mongodb://127.0.0.1:27017'
const databaseName = 'task-manager'

MongoClient.connect(connectionURL, { useNewUrlParser: true }, (error, client) =>
{
    if (error) {
        return console.log('Unable to connect to database’)
    }

    const db = client.db(databaseName)

    db.collection('users').updateOne(
        { name: 'Charles'},
        $set: {
            name: 'Mike’
        }
    )
});
```

Delete a document

```
const mongodb = require('mongodb')
const MongoClient = mongodb.MongoClient

const connectionURL = 'mongodb://127.0.0.1:27017'
const databaseName = 'task-manager'

MongoClient.connect(connectionURL, { useNewUrlParser: true }, (error, client) =>
{
    if (error) {
        return console.log('Unable to connect to database’)
    }

    const db = client.db(databaseName)

    db.collection('users’).deleteOne(
        { name: 'Charles’}
    ).then((result) => {}).catch((error) => {})
});
```

Exercices

- Intégrer une base de donnees dans notre application de taches.

Fin