



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...



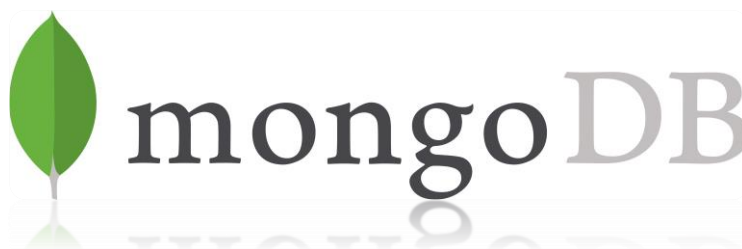
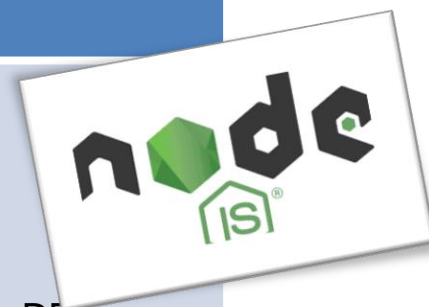
Plan du module Node.js

Cours	Date	Cours
1	Lun. 20/05	<ul style="list-style-type: none"> Introduction to Node.js (1)
2	Mer. 29/05	Building your Stack (2) <ul style="list-style-type: none"> Command Line File System Arrow Functions Modular JavaScript with Node.js
3	Mer. 05/06	<ul style="list-style-type: none"> Asynchronous JS Consuming API – HTTP requests
4	Lun. 10/06	<ul style="list-style-type: none"> Consuming API – HTTP requests Exercises – Notes & Weather
5	Lun. 17/06	<ul style="list-style-type: none"> 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	<ul style="list-style-type: none"> Promise, await, async, Async module ... Avoiding common pitfalls with Async.js
7	Lun 08/07	<ul style="list-style-type: none"> MongoDB ½
8	Lun 15/07	<ul style="list-style-type: none"> MongoDB 2/2
9	Lun 22/07	<ul style="list-style-type: none"> Working with MongoDB 1/2
10	Lun 15/07	<ul style="list-style-type: none"> Working with MongoDB 2/2



Module Node.js & MongoDB

Cours	Date	Cours
7	Lun. 08/07	<ul style="list-style-type: none">• 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	<ul style="list-style-type: none">• MongoDB Basics• API mongoose 1/2
9	Lun 22/07	<ul style="list-style-type: none">• API Mongoose 2/2• Async/Await integration with mongoDB
10	Lun 29/07	<ul style="list-style-type: none">• 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>



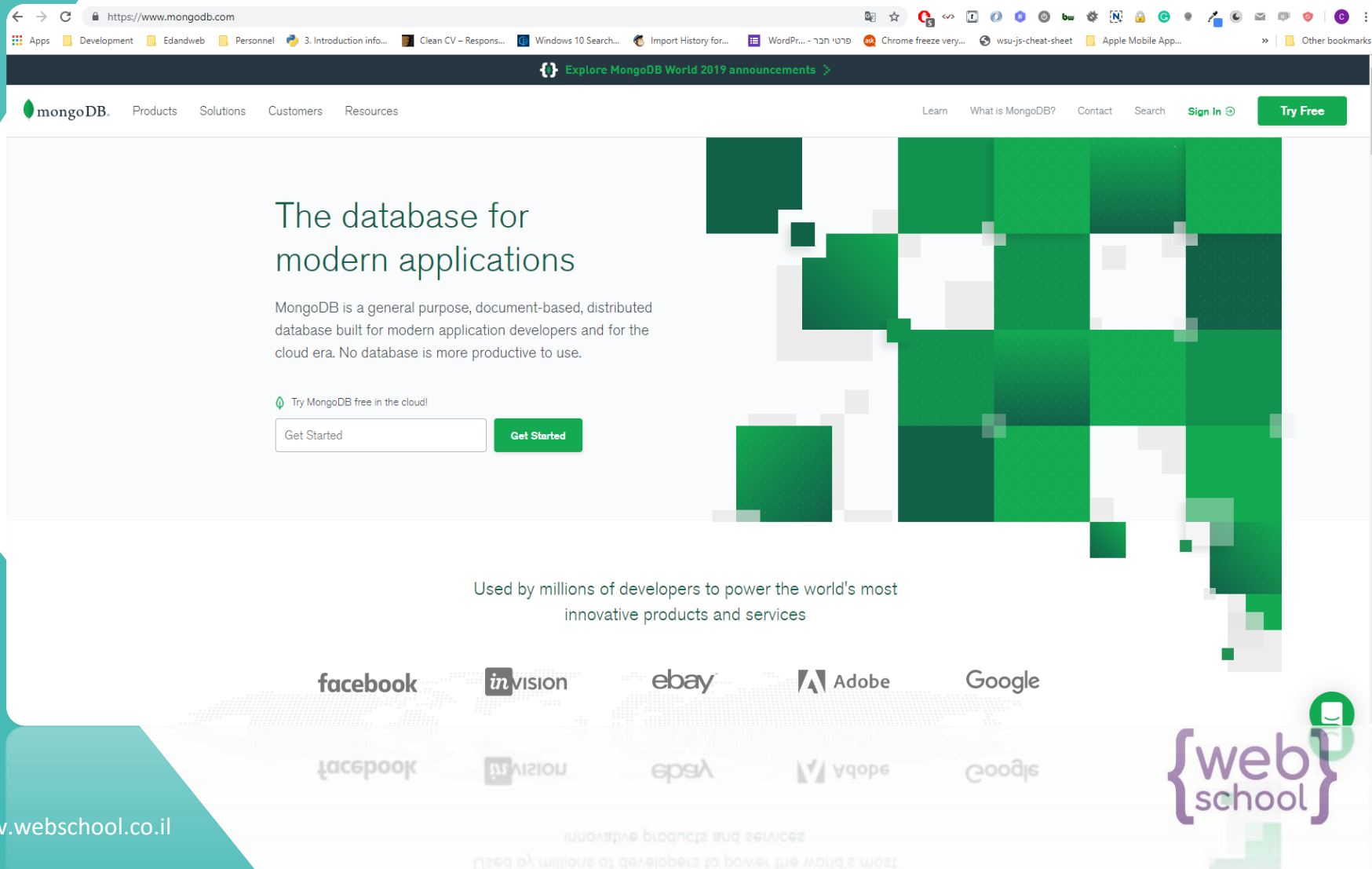
Résumé du cours 5

- 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}}`
`Nom: {{this.name}}, Age: {{this.age}}, Country: {{this.country}}`
`{{/each}}`
- Création d'un point d'entrée 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 fonction de l'adresse tapée dans le navigateur



Mongo DB

<http://www.mongodb.com>



The screenshot shows the MongoDB website homepage. The browser's address bar displays <https://www.mongodb.com>. The website's navigation bar includes the MongoDB logo, links for Products, Solutions, Customers, and Resources, and a 'Try Free' button. The main content area features the headline 'The database for modern applications' and a subtext describing MongoDB as a general purpose, document-based, distributed database. Below this, there is a 'Try MongoDB free in the cloud!' link and a 'Get Started' button. The footer section displays logos for various companies using MongoDB, including Facebook, InVision, eBay, Adobe, and Google. A 'web school' logo is visible in the bottom right corner.

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

The database for modern applications

MongoDB is a general purpose, document-based, distributed database built for modern application developers and for the cloud era. No database is more productive to use.

Try MongoDB free in the cloud!

Get Started Get Started

Used by millions of developers to power the world's most innovative products and services

facebook in vision ebay Adobe Google

facebook in vision ebay Adobe Google

{web} school

Introduction à MongoDB

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

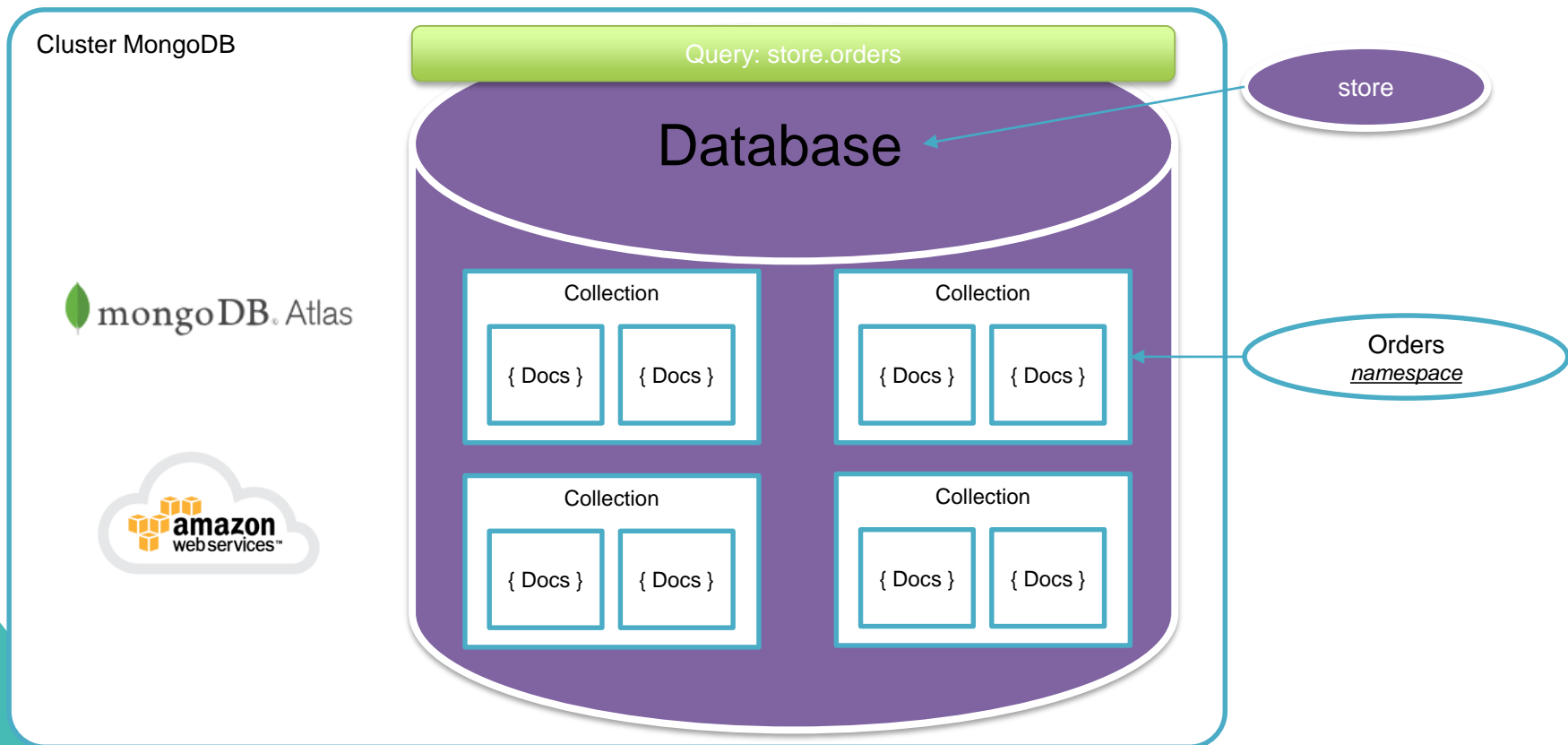


What is MongoDB ? (Wikipedia)

- **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

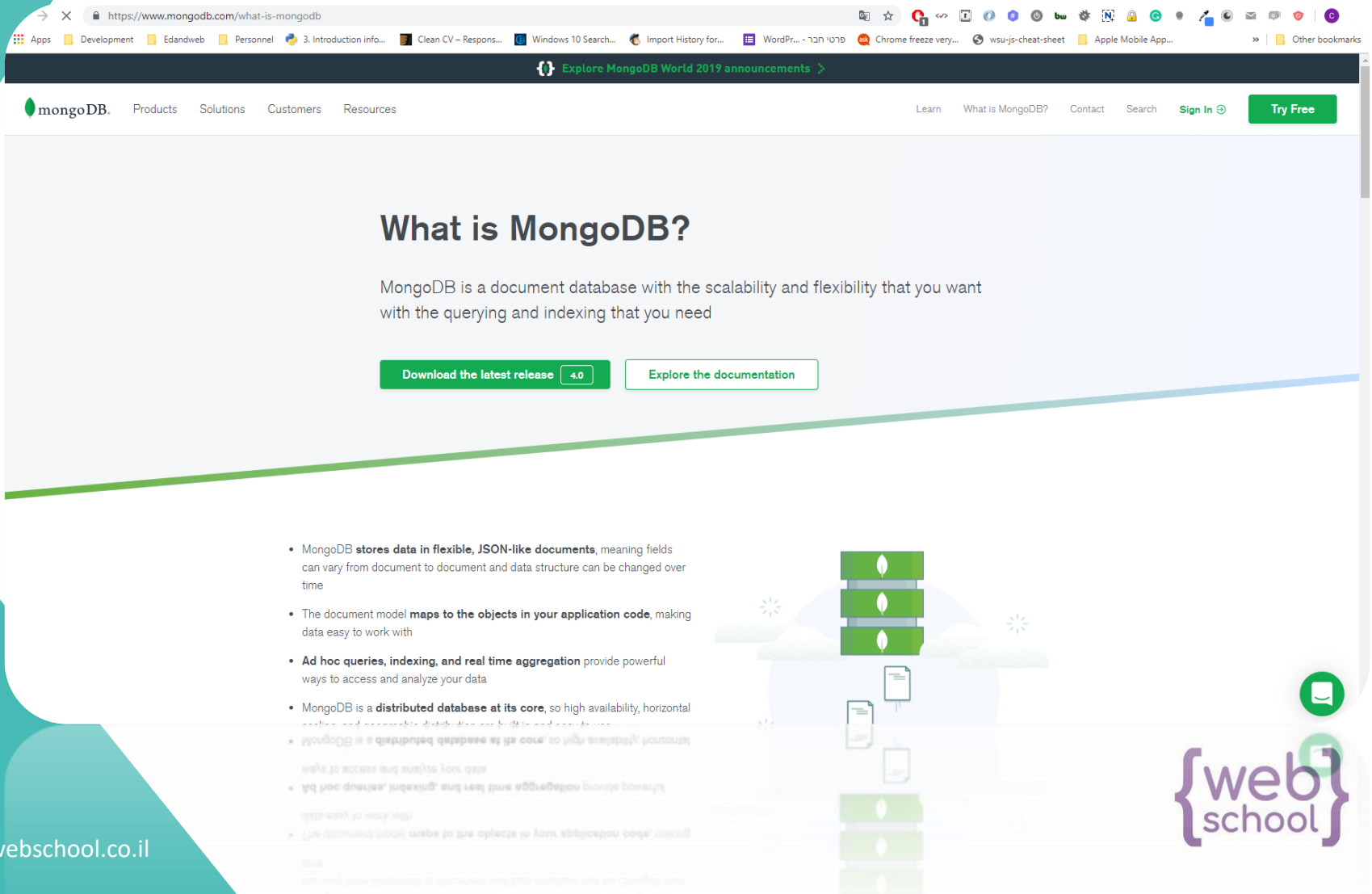


Database & GUI Installation

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



Installation serveur local



The screenshot shows the MongoDB website's 'What is MongoDB?' page. The browser's address bar displays 'https://www.mongodb.com/what-is-mongodb'. The website's navigation bar includes links for 'Products', 'Solutions', 'Customers', and 'Resources', along with a 'Try Free' button. The main heading is 'What is MongoDB?'. Below it, a paragraph states: 'MongoDB is a document database with the scalability and flexibility that you want with the querying and indexing that you need'. Two buttons are present: 'Download the latest release 4.0' and 'Explore the documentation'. A list of features is shown, including: 'MongoDB stores data in flexible, JSON-like documents', 'The document model maps to the objects in your application code', and 'Ad hoc queries, indexing, and real time aggregation provide powerful ways to access and analyze your data'. A decorative graphic on the right side of the page features a stack of green boxes with leaf icons, a cloud, and a document icon. The 'webschool' logo is visible in the bottom right corner.

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

Explore MongoDB World 2019 announcements >

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

What is MongoDB?

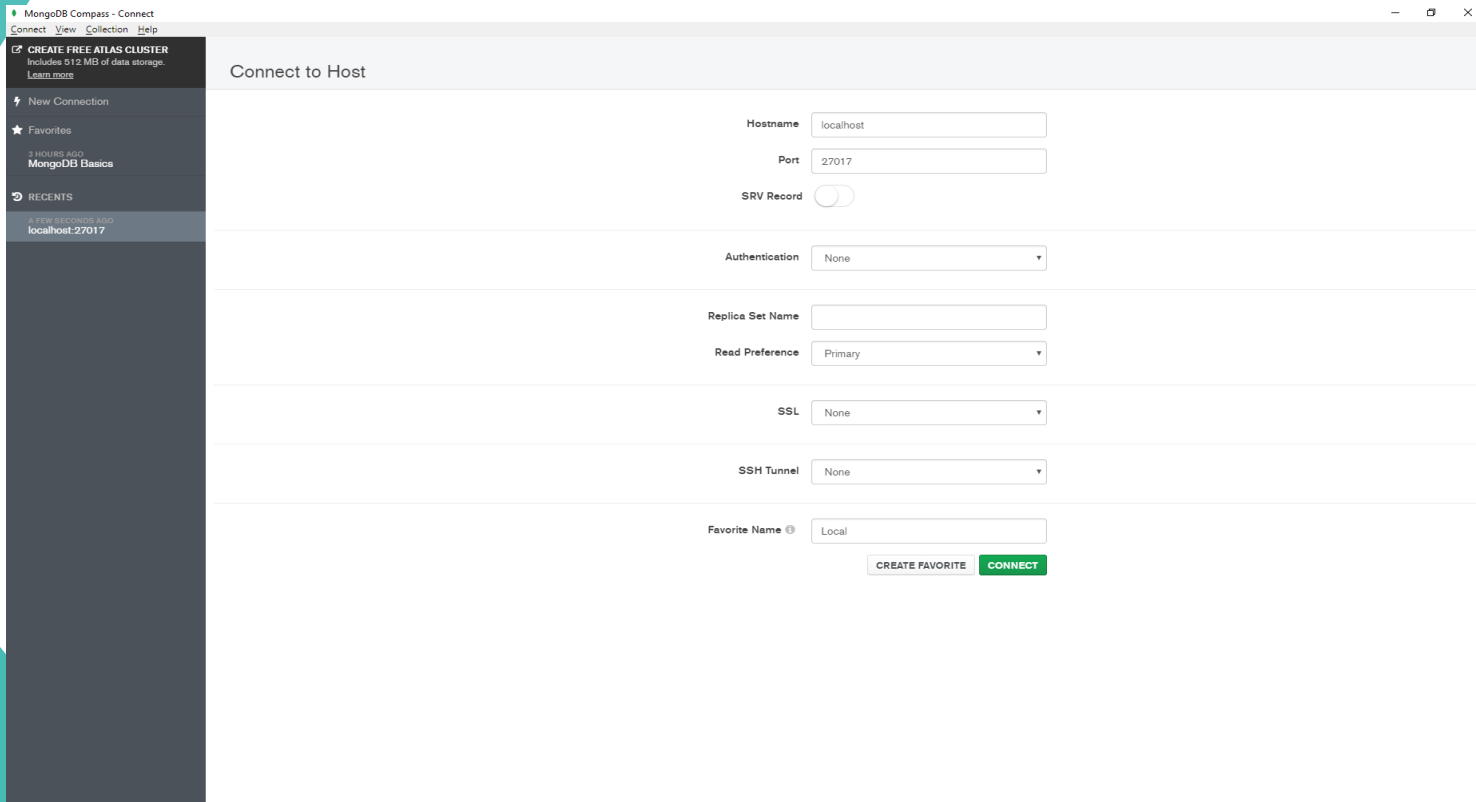
MongoDB is a document database with the scalability and flexibility that you want with the querying and indexing that you need

[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 scaling, and geographic distribution are built in and easy to use

[webschool](#)

Cluster - Connection



The screenshot shows the MongoDB Compass 'Connect' window. The left sidebar contains a 'CREATE FREE ATLAS CLUSTER' button, a 'New Connection' button, a 'Favorites' section with a 'MongoDB Basics' entry, and a 'RECENTS' section with a 'localhost:27017' entry. The main area is titled 'Connect to Host' and contains the following configuration fields:

- Hostname: localhost
- Port: 27017
- SRV Record: ☐
- Authentication: None
- Replica Set Name:
- Read Preference: Primary
- SSL: None
- SSH Tunnel: None
- Favorite Name: Local

At the bottom right of the main area are two buttons: 'CREATE FAVORITE' and 'CONNECT'.

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 official Node.js

The screenshot shows the MongoDB documentation page for the Node.js driver. The browser address bar displays the URL <https://docs.mongodb.com/ecosystem/drivers/node/>. The page has a navigation bar with tabs for SERVER, DRIVERS (active), CLOUD, TOOLS, and GUIDES. A search bar is located on the right. On the left, a sidebar titled 'MongoDB Ecosystem' lists various drivers, with 'Node.js Driver' highlighted. The main content area is titled 'MongoDB Node.js Driver' and includes a section 'On this page' with links to Introduction, Take the free online course taught by MongoDB, Installation, Connect to MongoDB Atlas, Compatibility, and How to get help. Below this is an 'Introduction' section stating that the official MongoDB Node.js driver provides both callback-based and Promise-based interaction with MongoDB. A list of links for Tutorials, Usage Guide, API Reference, Changelog, and Source Code is provided. A section titled 'Take the free online course taught by MongoDB' features a video player and a link to 'M220JS: MongoDB for JavaScript Developers'. The bottom right corner includes a 'Was this page helpful?' feedback widget and the 'webschool' logo.

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 חביר... Chrome freeze very... wsu-js-cheat-sheet Apple Mobile App... Other bookmarks

mongoDB. | Documentation

SERVER DRIVERS CLOUD TOOLS GUIDES Get MongoDB Search Documentation

MongoDB Ecosystem Close x

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?

{webschool}

www.webschool.co.il

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 données dans notre application de tâches.



Fin

