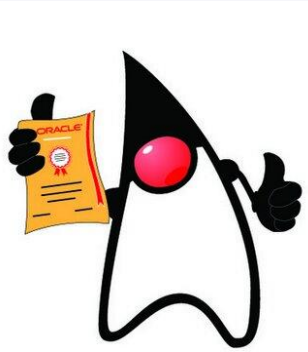
Maurisandev @MauriDeveloper mau.sanchez@bue.edu.ar

Curso Java FullStack

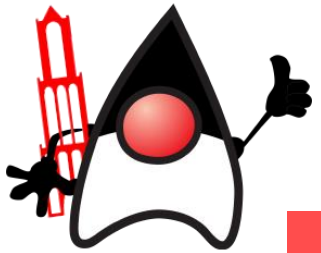
Codo a Codo 4.0

Clase-37–.

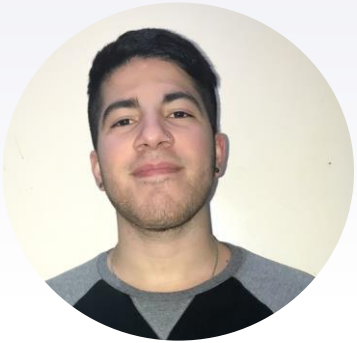
Cap. MONGODB



mongoDB.



Gracias por su aporte !!!!



Berbara Federico
Full Stack Developer
UADE

federicoberbara
feedee96@gmail.com
@fedeeberbara

Fretes Lautaro
Full Stack Developer
UADE

lautaro-fretes
lautarofretes55@gmail.com
@lautaro_fretes

01 | MONGODB

MongoDB

MongoDB es un sistema de base de datos NoSQL, orientado a documentos y de código abierto.

MongoDB

A diferencia de lo que ocurre en MySQL, donde tenemos tablas, registros, dentro de MongoDB vamos a tener colecciones, documentos y cada documento tiene una representación de clave-valor, donde clave es el nombre del campo y valor es el valor que puede tomar, entre los cuales vamos a poder encontrar enteros, strings, objetos, entre otros.

Un documento no es más que la representación de un objeto de la realidad.

MongoDB - Tipos de datos

STRING
name: String

```
{
  name: “John”
}
```

NUMBER
likes: Number

```
{
  likes: 5
}
```

DATE
timeStamp: Date

```
{
  timeStamp: ISODate(“...”)
}
```

ARRAY
tags: Array
OR
tags: []

```
{
  tags: [“tag1”, “tag2”]
}
```

BOOLEAN
published: Boolean

```
{
  published: true
}
```

ObjectId
_creator: Schema.ObjectId

```
{
  _creator: “41239878”
}
```

mongoDB.

MongoDB - Colecciones

Representan un conjunto de documentos de una misma entidad

Customer Collection

```
{
    _id: 123,
    name: "Katrina Pope",
    street: "123 Main St",
    city: "Somewhere",
    country: "Someplace",
    ...
}
```

MongoDB - En la nube

Vamos ahora a utilizar el servicio gratuito de MongoDb en la nube para poder tener nuestra primer DB no relacional. Para eso, nos registramos en mongodb.com. Una vez creados nuestro nodo, descargamos MongoDB Compass para poder conectarnos al mismo.

02 |SPIRNGBOOT +MONGODB

MongoDB – SringBoot

Technologies

- Java 11
- Spring Boot 2.2.4 (with Spring Web MVC, Spring Data MongoDB)
- MongoDB
- Maven 3.6.1

MongoDB – SringBoot

Para configurar el proyecto Spring Boot

Utilice la herramienta web Spring o su herramienta de desarrollo (Spring Tool Suite , Eclipse, Intellij) para crear un proyecto Spring Boot.

Luego pom.xml y agregue estas dependencias:

```
<<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-mongodb</artifactId>
</dependency
```

Configurar Spring Data MongoDB

En la carpeta src / main / resources , abra application.properties y agregue las siguientes líneas.

```
#mongodb
spring.data.mongodb.host=localhost
spring.data.mongodb.port=27017
spring.data.mongodb.database=ecommerce
```


```java
package com.maudev.springboot.app.models;

import java.io.Serializable;

import org.springframework.data.mongodb.core.mapping.Document;
import org.springframework.data.mongodb.core.mapping.FieldType;
import org.springframework.data.mongodb.core.mapping.MongoId;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@NoArgsConstructor
@AllArgsConstructor
@Data
@Document(collection = "productos")
public class Producto implements Serializable {

    private static final long serialVersionUID = 1731902390885211826L;

    @MongoId(value = FieldType.OBJECT_ID)
    private String id;

    private String nombre;

    private String descripcion;

    private String marca;

    private Double precio;

    private Integer cantidad;

    private String rubro;

}
```


```java
package com.maudev.springboot.app.repository;

import java.io.Serializable;

import org.springframework.data.mongodb.repository.MongoRepository;
import org.springframework.stereotype.Repository;

import com.maudev.springboot.app.models.Producto;

@Repository("ProductoRepository")
public interface ProductoRepository extends MongoRepository<Producto, Serializable> {
    // DENTRO DE ESTE BLOQUE DE CODIGO VOY A HACER LAS CONSULTAS
    // QUE NECESITE CONTRA MONGO DB

    // internamente Actua como una Querry de consulta
    // Busca en base de datos un producto por medio del ID
    Producto findById(String id);

}
```


```java
package com.maudev.springboot.app.services;

import com.maudev.springboot.app.models.Producto;


public interface IProductoService {

    //metodo interrface para poder utilizar tanto en Service implementacion
    //Como tambien en el Controller
    public abstract Producto IdProducto(String idProducto);
}
```

mongoDB.


```java
package com.maudev.springboot.app.services.impl;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.maudev.springboot.app.models.Producto;
import com.maudev.springboot.app.repository.ProductoRepository;
import com.maudev.springboot.app.services.IProductoService;

@Service("productoService")
public class ProductoServiceImpl implements IProductoService {

    // Inyecto el Repository
    // Para usar sus funciones e implementar
    // en la logica de los sevicios
    @Autowired
    private ProductoRepository productoRepository;

    // con este contrato o metodo de mi capa actual
    // creo la logica para poder hacer la consulta con mi base de datos monngo
    @Override
    public Producto IdProducto(String idProducto) {
        // Instancio un nuevo objeto producto
        Producto producto = null;

        // Envolvemos en una validacion
        // para que en caso falle por algun motivo esa consulta
        // me imprima en la consola en que parte fallo mi app
        try {

            producto = productoRepository.findById(idProducto);

        } catch (Exception e) {

            e.printStackTrace();
        }

        // En caso de exito devuelvo todos los datos que contiene Producto
        return producto;

    }

}
```


```java
package com.maudev.springboot.app.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.MediaType;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RestController;

import com.maudev.springboot.app.models.Producto;
import com.maudev.springboot.app.services.IProductoService;

@RestController
public class ProductoController {

    @Autowired
    private IProductoService productoService;

    // Metodo GET para obtener los datos de 1 producto por su ID
    @GetMapping(value = "/producto/{idProducto}", produces = MediaType.APPLICATION_JSON_VALUE)
    public ResponseEntity<Producto> findByProducto(@PathVariable String idProducto) {

        // Instancio un nuevo objeto producto
        Producto producto = new Producto();

        // llamor al servicio creado y le paso por parametro el idProducto
        producto = productoService.IdProducto(idProducto);

        // Si el servicio me devolvio un resultado exitoso o 200
        // devuelvo al FrontEnd todos los datos del Producto solicitado
        return ResponseEntity.ok(producto);
    }

}
```


```java
package com.maudev.springboot.app.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

import springfox.documentation.builders.PathSelectors;
import springfox.documentation.builders.RequestHandlerSelectors;
import springfox.documentation.spi.DocumentationType;
import springfox.documentation.spring.web.plugins.Docket;
import springfox.documentation.swagger2.annotations.EnableSwagger2;

@Configuration
@EnableSwagger2
public class SprigFoxConfig {

    //Configuracion de SWAGGER
    @Bean
    public Docket api() {
        return new
Docket(DocumentationType.SWAGGER_2).select().apis(RequestHandlerSelectors.any())
                .paths(PathSelectors.any()).build();
    }

}
```

DBA (Relacionales y No Relacionales)

DBA (Relacionales/No Relacionales) Diferencias

Una *base de datos relacional* es un tipo de base de datos que almacena y proporciona acceso a puntos de datos relacionados entre sí. Las bases de datos relacionales se basan en el modelo relacional, una forma intuitiva y directa de representar datos en tablas.

Una base de datos no relacional Están diseñadas específicamente para modelos de datos específicos y tienen esquemas flexibles para crear aplicaciones modernas. Son ampliamente reconocidas porque son fáciles de desarrollar, tanto en funcionalidad como en rendimiento a escala.

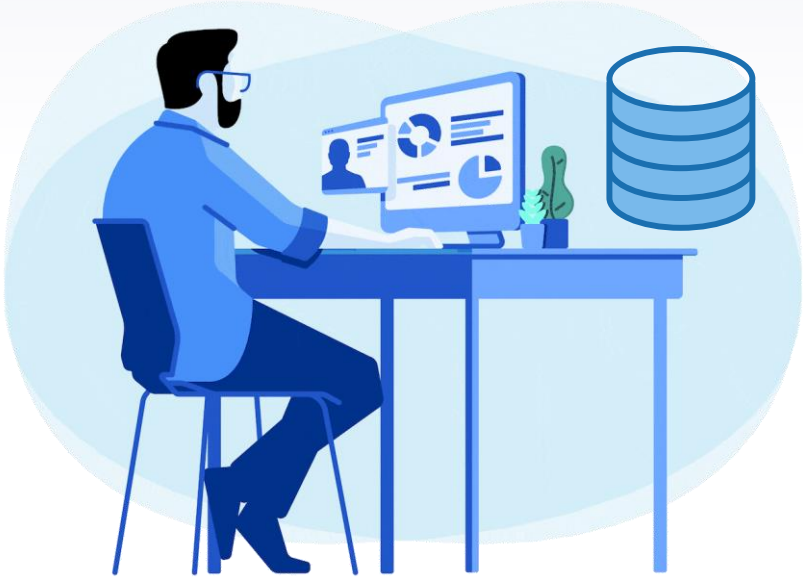
<codoa codo/>

Instalación MongoDB (NO RELACIONAL)

- https://www.mongodb.com/
- Tutorial:
- https://www.codewall.co.uk/mongodb-beginner-tutorial-with-compass-gui-the-mongo-shell-cli/

Maurisandev @MauriDeveloper maurisan4011@gmail.com

Muchas Gracias!



****

NOS VEMOS EL
Jueves 19:00!!

<codoa codo/>