**M165 Pokémon Quiz Dokumentation**

Inhaltsverzeichnis

[Unser Thema 2](#_Toc154736057)

[DB erstellen 4](#_Toc154736058)

[DB testen 4](#_Toc154736059)

[DB werte lesen in Java 5](#_Toc154736060)

[Datenbank durchlaufen 6](#_Toc154736061)

[Von BSON zu Java 6](#_Toc154736062)

## Unser Thema

Wir haben uns entschieden für das Quiz Pokémon Karten als Vorlage zu haben wo wir als Messbare Werte:

1. HP, Healhpoints
2. DMG, Strength
3. Energy Points
4. Stage, Evolution Stage

Beispielsweisse wird dann für das Quiz die Frage kommen für welche Kategorie man befragt werden will, man gibt die Antwort im Terminal ein und bekommt eine Frage mit 3 Pokémons zur Auswahl.

Wir machen mal ein Beispiel Lauf

Ein Bild, das Text, Schrift, Screenshot enthält.

Automatisch generierte Beschreibung

wir wählen die Kategorie «Strength»

Ein Bild, das Text, Schrift, Screenshot, Algebra enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte BeschreibungWir bekommen jetzt diese 3 zufällige Pokémons zur Auswahl:

  

! im Spiel hat man keine Bilder zu den Pokémons !

Und wir geben unsere Antwort:

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Nachdem wir die Antwort geben, wird es aufgelöst, ob wir richtig oder falsch sind und welches die richtige Antwort ist.

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

## DB erstellen

Mit einem config file erstellen wir die Anforderungen für unsere DB, welche eine ganz normale MongoDB ist, die schon mit werten voll ist welche es von einem init.js File die Objekte für die Collection abliesst.

Ein Bild, das Text, Screenshot, Software enthält.

Automatisch generierte Beschreibung Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte Beschreibung

### DB testen

Diese Daten der Pokémons sind jetzt auf unserer PokémonDB gespeichert und das können wir über InteliJ oder MongoDB Compass testen.

Ein Bild, das Text, Screenshot, Software, Zahl enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text, Screenshot, Software, Display enthält.

Automatisch generierte Beschreibung

## DB werte lesen in Java

Um jetzt mit den Pokémons aus der Datenbank arbeiten zu können, muss man die Werte mit Java einlesen. Wir haben als Vorbereitung eine Pokémon Klasse erstellt, um eine Vorlage für die Pokémons in der Datenbank zu haben. Zum Verbinden zur Datenbank haben wir die Mongodb-driver dependency genutzt, welche es uns erlaubt sich mit einer Mongo Datenbank zu verbinden.

String connectionString = "mongodb://root:1234@localhost:27017/?authSource=admin";

Mithilfe der Datenbank URI welche sowohl den Username als auch Password beinhaltete konnten wir uns erfolgreich mit dem MongoClient verbinden. Mit dem Mongoclient verbunden, wählten wir unsere Datenbank und unsere Pokémon collection.

// Connecting to database & get collection  
try (MongoClient mongoClient = MongoClients.*create*(connectionString)) {  
 MongoDatabase database = mongoClient.getDatabase("mongodb");  
 MongoCollection<Document> collection = database.getCollection("pokemons");

### Datenbank durchlaufen

Um drei zufällige Pokémons zu erhalten, wird bei der collection die aggregation "$sample" verwendet. Mit dem Nutzen dieser aggregation ist es möglich, die Anzahl der zufälligen Pokémons zu bestimmen.

var aggregation = Arrays.*asList*(Aggregates.*sample*(3));

Die Werte werden nun in ein BSON Document hinzugefügt. In unserem Fall die drei zufällige Pokémons. Zuerst wird ein "cursor" erstellt, welchem die drei Documents übergeben werden.

### Von BSON zu Java

Nun da man die Werte des Pokémons in einem BSON Document zur verfügung hat, kann man diese in ein Java Objekt umwandeln. Dazu wird in der Methode das Document als Parameter mitgegeben. Mit den Settern werden die Werte gesetzt und ein fertiges Pokémon Objekt ist bereit.

private Pokémon documentToPokemon(Document document) {

Pokémon pokemon = new Pokémon();

pokemon.setName(document.getString("name"));

pokemon.setHealth(document.getInteger("health"));

pokemon.setStrenght(document.getInteger("strenght"));

pokemon.setEnergy(document.getInteger("energy"));

pokemon.setStage(document.getInteger("stage"));

return pokemon;

}