

Aquarium 2.0 Datenbankzugriffsmodul (DAO)

Beschreibung

Dieses Modul soll eine Kommunikation zwischen Datenbank und anderen Programm-Modulen ermöglichen.

Kommunikationseinstellungen (Connection Properties)

Kommunikationseinstellungen mit der Datenbank werden aus Properties-Datei gelesen. Kommunikationseinstellungen bestehen aus folgenden Strings:

- Typ (MySQL, Oracle)
- Host (localhost, 192.168.0.1)
- Port (3306, 1521)
- DBname (aqua)
- DBuser (aqua)
- DBpass (wifi)

Datenbankabfragen (Statements)

Datenbankabfragen werden von GUI- und Upload-Modulen durch Methodenaufruf initiiert. Bei jeder Abfrage wird eine Connection aufgebaut, vorbereitete Abfragen (Prepared Statements) werden ausgeführt, Daten aus ResultSet in benötigte Objekte (Strings, Arrays, ArrayLists, Vectors) umgewandelt und zum Aufrufsmodul zurück gegeben.

Ausnahmen (Exeptions)

Beim Aufbauen der Verbindung und den Abfragen werden SQL- und andere Ausnahmen (Exceptions) abgefangen, protokolliert(Log) und zum Aufrufsmodul als DAO-Exceptions wieder geworfen.

Entwurfsmuster (Design Pattern)

Data Access Object (DAO, deutsch: „Datenzugriffsobjekt“) ist ein Entwurfsmuster, das den Zugriff auf unterschiedliche Arten von Datenquellen (z. B. Datenbanken, Dateisystem, etc.) so kapselt, dass die angesprochene Datenquelle ausgetauscht werden kann, ohne den aufrufenden Code zu ändern. Dadurch soll die eigentliche Programmlogik von technischen Details der Datenspeicherung befreit werden und flexibler einsetzbar sein.

Wikipedia, http://de.wikipedia.org/wiki/Data_Access_Object

```

sequenceDiagram
    participant D as :DimensionenList
    participant A as :AquaDaoFactory
    participant AM as :AquaDaoFactoryMysql
    participant DM as :DimensionenDaoMysql

    D->>A: getInstance().getDimensionenDao().read()
    activate A
    A->>A: loadProperties()
    A->>AM: new AquaDaoFactoryMysql(properties: Properties)
    activate AM
    AM->>DM: new DimensionenDaoMysql(uri: String)
    activate DM
    DM->>DM: read()
    DM->>DM: createDimensionenList(rs: ResultSet): Vector<Dimensionen>
    deactivate DM
    AM-->>A: 
    deactivate AM
    A-->>D: ~Vector<Dimensionen>
    deactivate A
  
```

class AquaDAOclass

EA 7.1 Unregistered Trial Version

DAO

AquaDaoFactory

- PROPERTIES_FILE: String (readOnly)
- MYSQL_DRIVER: String (readOnly)
- instance: AquaDaoFactory

+ getInstance(): AquaDaoFactory
 + getStatistikDao(): StatistikDao
 + getSzenarienDao(): SzenarienDao
 + getSzenarioFutterzeitDao(): SzenarioFutterzeitDao
 + getDimensionenDao(): DimensionenDao
 + getLebewesenDao(): LebewesenDao
 + getPhwerteDao(): PhwerteDao
 + getTempwerteDao(): TempwerteDao

AquaDaoFactoryMysql

- url: String
- statistikDao: StatistikDao
- szenarienDao: SzenarienDao
- szenarioFutterzeitDao: SzenarioFutterzeitDao
- dimensionenDao: DimensionenDao
- lebewesenDao: LebewesenDao
- phwerteDao: PhwerteDao
- tempwerteDao: TempwerteDao

AquaDaoMysql

- url: String
- c: Connection
- s: Statement
- ps: PreparedStatement
- cs: CallableStatement

+ close(): void
 + getConnection(): Connection
 + getStatement(): Statement
 + getPreparedStatement(String): PreparedStatement
 + getCallableStatement(String): CallableStatement

DimensionenDaoMysql

- + INSERT: String (readOnly)
- + SELECT_ALL: String (readOnly)
- + UPDATE: String (readOnly)
- + DELETE_BY_ID: String (readOnly)

«interface» DimensionenDao

- + create(int): int[]
- + read(): Vector<Dimensionen>
- + update(int, int[]): void
- + delete(int): void

View

PanelSzenario

- aquaPanel: PanelSzenarioCombo
- dimPanel: PanelDimensionen
- aquaModel: DimensionenList

+ createAquarium(): void
 + editAquarium(): void
 + deleteAquarium(): void

Model

DimensionenList

- dao: DimensionenDao

+ create(): Dimensionen
 + save(Object): void
 + delete(int): void

Dimensionen

- id: int
- values: int[]

+ toString(): String
 + getId(): int
 + getValues(): int[]
 + setValues(Object, Object, Object): void
 + save(DimensionenDao): int
 + delete(DimensionenDao): void

Data Access Object (DAO, deutsch: „Datenzugriffsobjekt“) ist ein Entwurfsmuster, das den Zugriff auf unterschiedliche Arten von Datenquellen (z. B. Datenbanken, Dateisystem, etc.) so kapselt, dass die angesprochene Datenquelle ausgetauscht werden kann, ohne den aufrufenden Code zu ändern. Dadurch soll die eigentliche Programmlogik von technischen Details der Datenspeicherung befreit werden und flexibler einsetzbar sein.

Wikipedia, http://de.wikipedia.org/wiki/Data_Access_Object

EA 7.1 Unregistered Trial Version