
Ktor


OPEN
VALUE

Introduction

- framework for building web applications and microservices in Kotlin
- created by JetBrains
- Open Source
- written in Kotlin
  - Java compatibility

Architecture

- lightweight
  - only include the components you need
  - small learning curve
- asynchronous
- build on top of coroutines
- DSL for routing and handling requests

Where to use

- small to medium projects
  - smaller team
- microservices with IO load
- smaller CRUD apps
- realtime apps

Example Code

```
fun Application.module() {
    install(ContentNegotiation)
    routing {
        get("/") {
            call.respond(HttpStatusCode.OK, "Hello, Ktor!")
        }
    }
}

fun main(args: Array<String>) {
    io.ktor.server.netty.EngineMain.main(args)
}
```

Resources

- https://ktor.io/
- https://start.ktor.io/

EXPOSED

Introduction

- ORM framework for Kotlin
- created by JetBrains
- Open Source
- written in Kotlin

Architecture

- built on top of a JDBC driver
- two flavors of database access
  - typesafe SQL wrapping DSL
    - Kotlin centric way
  - lightweight Data Access Objects (DAO)
    - closer to JPA

Supported Databases

- H2
- MariaDB
- MySQL
- Oracle
- PostgreSQL
- Microsoft SQL Server
- SQLite

Example Code - DSL

- defining a table

```
object Users: Table() {
  val id = integer("id").autoIncrement()
  val name = varchar("name", 255).index()
  override val primaryKey = PrimaryKey(id)
}
```

Example Code - DSL

- querying the table using the DSL

```
fun exists(username: String): Boolean =
  Users.selectAll()
       .where { (name eq username) }
       .any()
```

Example Code - DAO

- defining table and an an entity

```
object UsersTable : IntIdTable() {
  val name = varchar("name", 255).uniqueIndex()
}
```

```
class UserEntity(id: EntityID<Int>) : IntEntity(id) {
    companion object : IntEntityClass<UserEntity>(UsersTable)

    var name by UsersTable.name
}
```

Example Code - DAO

- querying the table using a DAO

```
val user = UserEntity.new("Alice")

val allUsers = UserEntity.all()

val alice = UserEntity.find{ UsersTable.name eq "Alice" }
```

Where to use

- small to medium-sized Kotlin applications
- prototyping and rapid development
- applications requiring fine-grained database control
  - DSL close to actual queries

Resources

- https://github.com/JetBrains/Exposed
- https://jetbrains.github.io/Exposed/home.html