Pure Kotlin backend development with Ktor

Agenda

What is Ktor ?

- DSL Server / Client
- Multiplatform
- Asynchronous
- Pure Kotlin

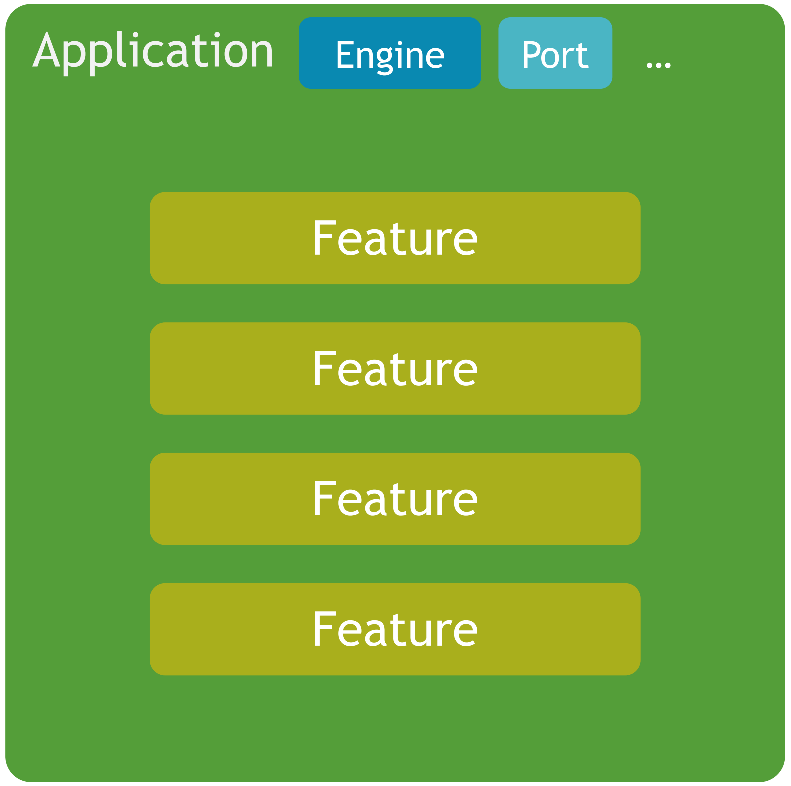
What is Ktor for ?

- *Building server applications*
  - *HTTP APIs*
  - *REST systems*
  - *(Web)Sockets*
- *Building client connections*

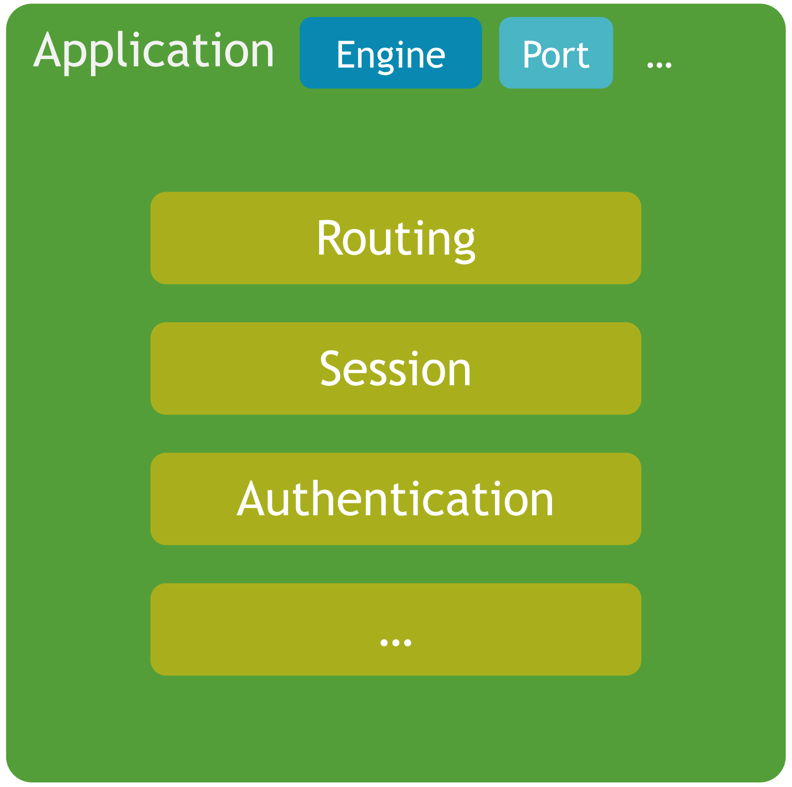
Ktor *on the backend*

```
fun main() {
    val server = embeddedServer(Netty, port = 8080) {
        routing {
            get("/") {
                call.respondText("Hello World!", ContentType.Text.Plain)
            }
        }
    }
    server.start(wait = true)
}
```

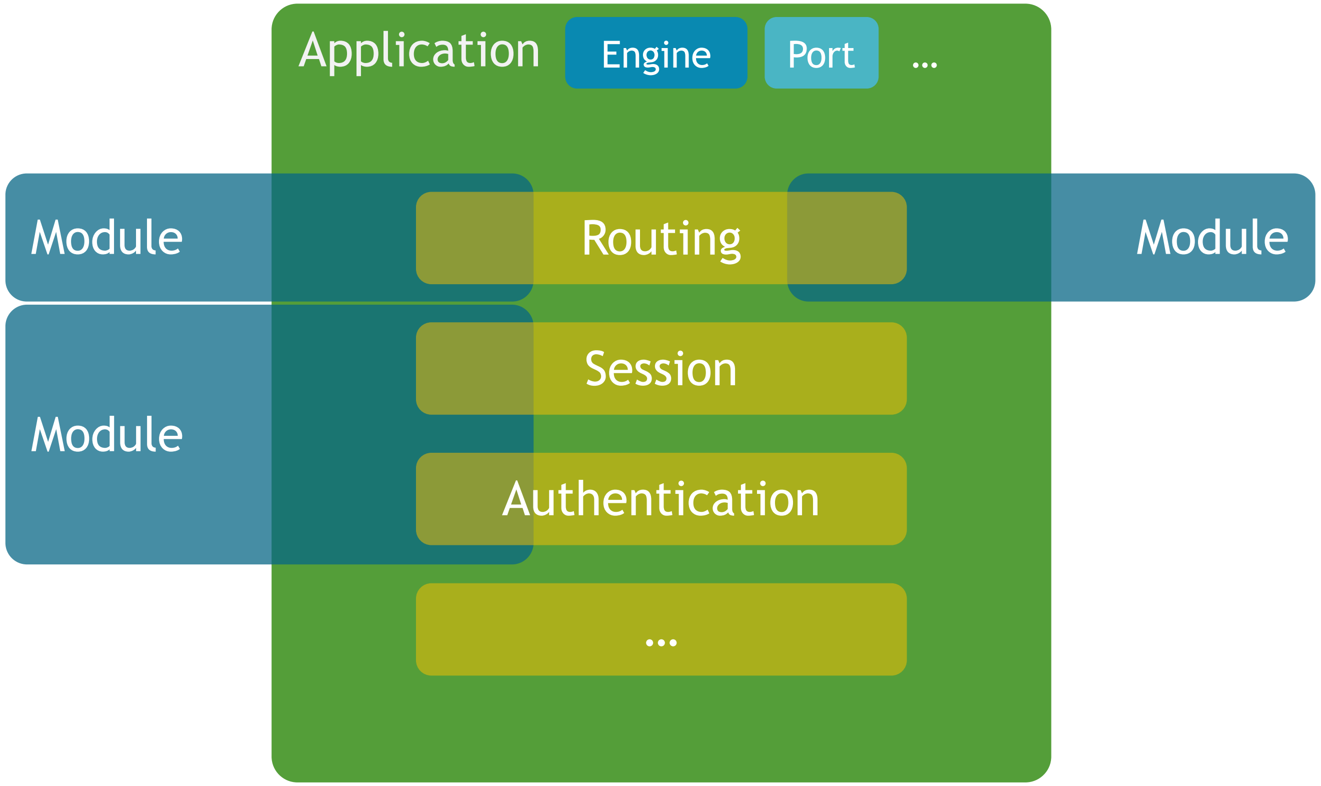
Ktor *on the backend*



Ktor *on the backend*



Ktor *on the backend*



Show us some code !

What is Kodein ?

- Stands for **KO**tlin **DE**pendency **IN**jection
- DI Library as part of a framework
- Pure Kotlin
- Multiplatform

Kodein extensions for Ktor

- Custom Feature

```
fun Application.module() {
    install(KodeinFeature) {
        bind<UserService>() with singleton { UserService() }
        bind<ItemService>() with singleton { BasicItemService() }
    }
}
```

Kodein extensions for Ktor

- Custom Feature

```
fun Application.module() {
    kodein {
        bind<UserService>() with singleton { UserService() }
        bind<ItemService>() with singleton { BasicItemService() }
    }
}
```

Kodein extensions for Ktor

- Custom Feature
- Closest Kodein pattern

```
fun Application.module() {
    install(KodeinFeature) {
        bind<ItemService>() with singleton { BasicItemService() }
    }

    routing {
        get("/todolist") {
            val itemService: ItemService by kodein().instance()
            // logic here
        }
    }
}
```

Kodein extensions for Ktor

- Custom Feature
- Closest Kodein pattern

- Scoping on Ktor objects
  - SessionScope
  - CallScope

Kodein extensions for Ktor

- Controller, a MVC-like architecture

```
class MyController(application: Application) : KodeinController {
    override val kodein by kodein { application }
    private val repository: DataRepository by instance("dao")

    override fun Routing.installRoutes() {
        get("/version") {
            val version: String by instance("version")
            call.respondText(version)
        }
    }
```

Kodein extensions for Ktor

- Controller, a MVC-like architecture

```
fun Application.module() {
    kodein {
        bind<MyController>() with singleton { MyController(instance()) }
    }
    install(KodeinControllerFeature)
}
```

Show us some code !

Thanks

Any questions ?

https://ktor.io/

http://kodein.org/

github: /romainbsl/ktor-on-kodein-talk