DEMYSTIFYING THE SERVER FOR MOBILE DEVS WITH KTOR

ALEJANDRO RIOS

CONDOR LABS

#KotlinConfGlobal2020

🧐 Dato curioso

🧐 Dato curioso

Puede parecer seguro asumir que las Islas Canarias llevan el nombre de las aves canarias, pero el conjunto de islas en realidad lleva el nombre de los perros. Aunque está frente a la costa del noroeste de África, el archipiélago es en realidad parte de España. En español, el nombre del área es Islas Canarias, que proviene de la frase latina ***Canariae Insulae*** que significa "isla de perros".

https://bestlifeonline.com/world-facts/

Entonces……
servidores??

https://i.ya-webdesign.com/images/server-icon-png-12.png

Código del lado del servidor, Yo?

https://thumbs.gfycat.com/GivingLastingIbadanmalimbe-size_restricted.gif

Servidores

- Existen muchas formas de crearlos, cuál escojo?

Servidores

- Existen muchas formas de crearlos, cuál escojo?
- Debo aprender un lenguaje diferente?

Servidores

- Existen muchas formas de crearlos, cuál escojo?
- Debo aprender un lenguaje diferente?
- Debo cambiar mi entorno de trabajo?

https://media.giphy.com/media/8lQyyys3SGBoUUxrUp/source.gif

Ktor! 🎉🎉🎉🎉🎉🎉🎉🎉

https://www.scottbrady91.com/img/logos/ktor.png

Ktor

- Se pronuncia *Kay-tor*

Ktor

- Se pronuncia *Kay-tor*
- Escrita en Kotlin

Ktor

- Se pronuncia *Kay-tor*
- Escrita en Kotlin
- Ligera y directa

Ktor

- Se pronuncia *Kay-tor*
- Escrita en Kotlin
- Ligera y directa
- Usa Corutinas

Ktor

- Se pronuncia *Kay-tor*
- Escrita en Kotlin
- Ligera y directa
- Usa Corutinas
- Lenguaje DSL

😐 Qué es lo que vamos a hacer???

Un "GitHub Dashboard" basico

- Un "GitHub Dashboard" basico

- Un "GitHub Dashboard" basico
- RESTful API

- Un "GitHub Dashboard" basico
- RESTful API
- WebApp

Empecemos construyendo el API RESTful!

🤔 Que vamos a usar?

Ktor

Ktor
OAuth2

Ktor
OAuth2
Retrofit

Ktor
OAuth2
Retrofit
Gson

Ktor
OAuth2
Retrofit
Gson
GitHub APIs

🙃 Cosas de las que no vamos a hablar

😕 Nope!

(se los dejo de tarea)

Quickstart
Sesiones
Bases de datos
Testing

https://chumley.barstoolsports.com/wp-content/uploads/2018/07/12/5555.gif

Ktor - estructura basica

ApplicationEngine

Tal como suena, es el motor, el proceso que se encarga de ejecutar el ambiente y correr la aplicación

CIO

Jetty

Netty

Tomcat

Coroutine-based I/O

Netty

CIO

Jetty

Tomcat

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

ApplicationEnvironment

application.conf

```
ktor {
  deployment {
    port = 8080
    watch = [Dev/kotlin/ktor/src]
  }
  application {
    modules = [com.alejandrorios.githubktor.ApplicationKt.module]
  }
}
```

```
ktor {
  deployment {
    port = 8080
    watch = [Dev/kotlin/ktor/src]
  }
  application {
     modules = [com.alejandrorios.githubktor.ApplicationKt.module]
  }
}
```

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

fun Application.module() {

}
```

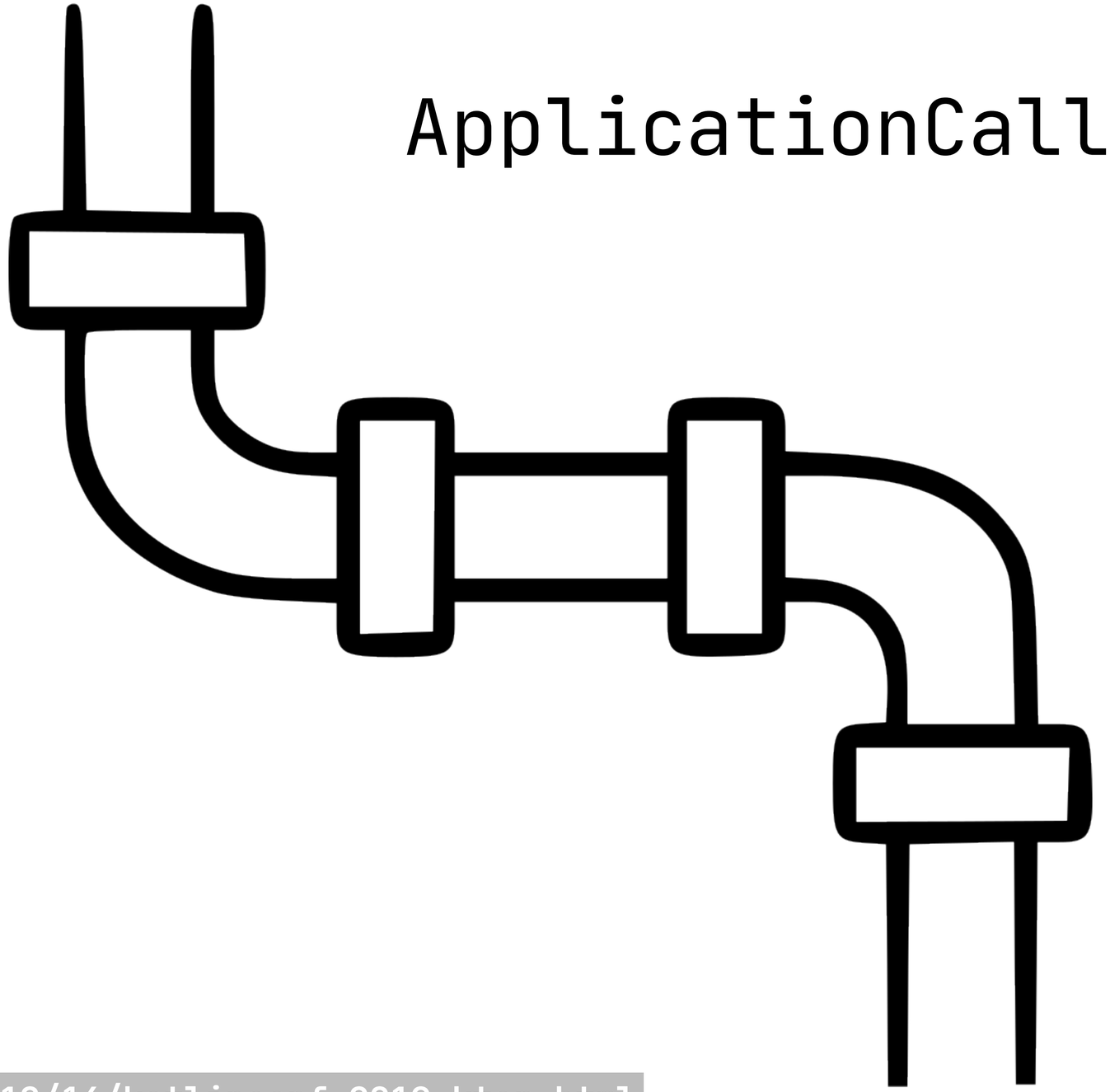
Application

```
class Application(val environment: ApplicationEnvironment) :
  ApplicationCallPipeline(), CoroutineScope {

   ...
}
```

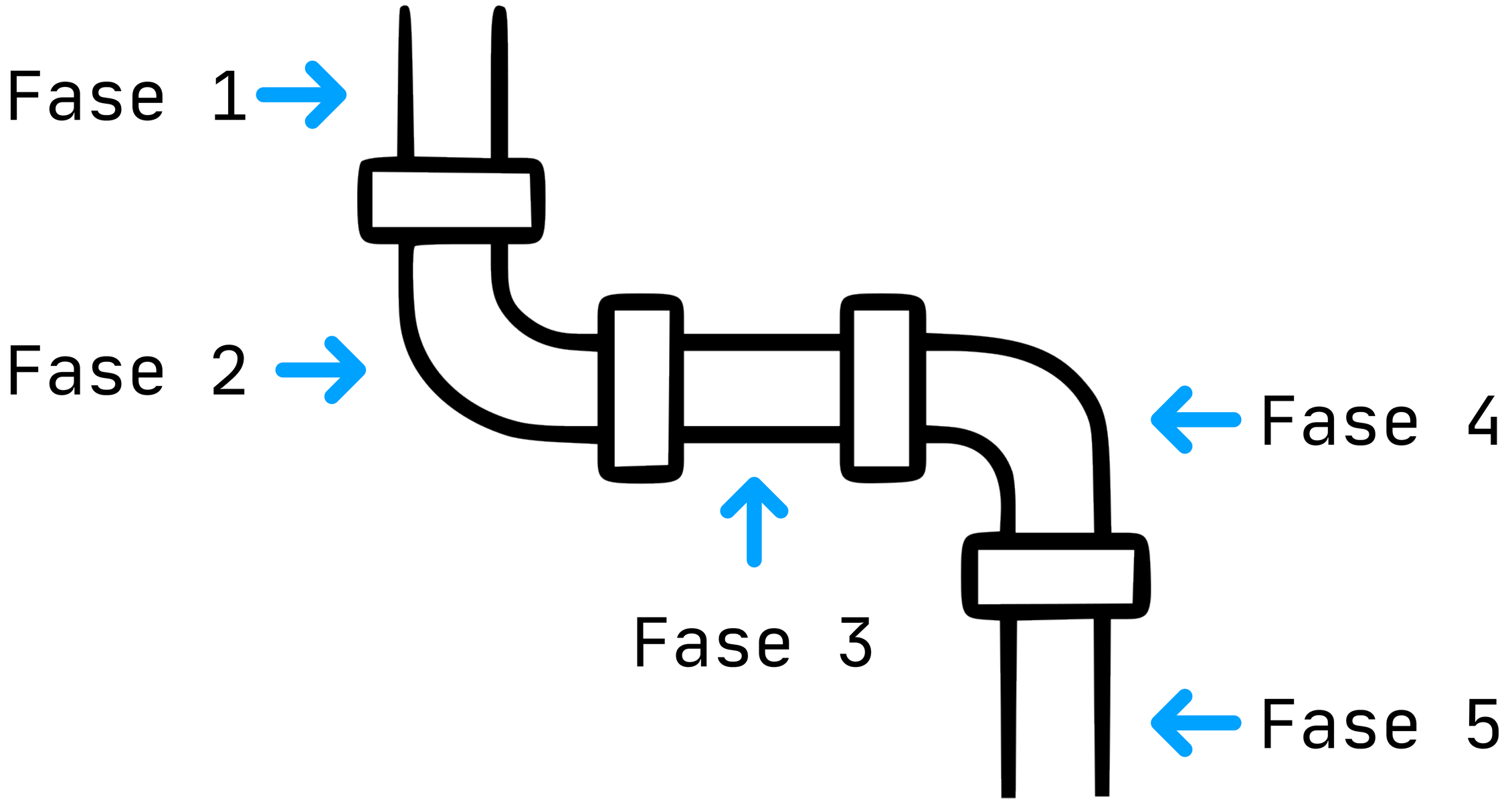
```
class Application(val environment: ApplicationEnvironment) :
    ApplicationCallPipeline(), CoroutineScope {

    ...
}
```

ApplicationCallPipeline

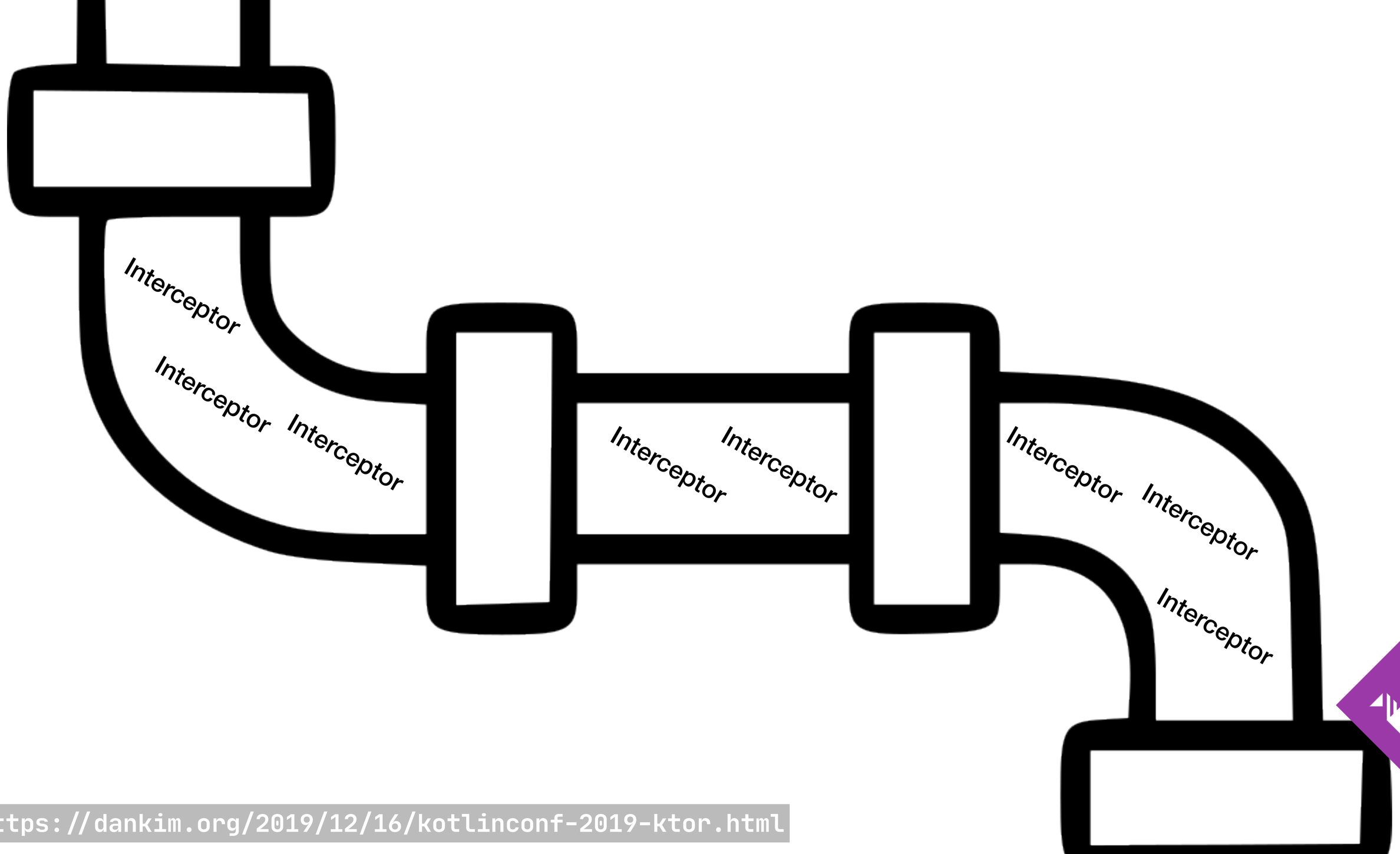
ApplicationCallPipeline



https://dankim.org/2019/12/16/kotlinconf-2019-ktor.html



https://dankim.org/2019/12/16/kotlinconf-2019-ktor.html



https://dankim.org/2019/12/16/kotlinconf-2019-ktor.html

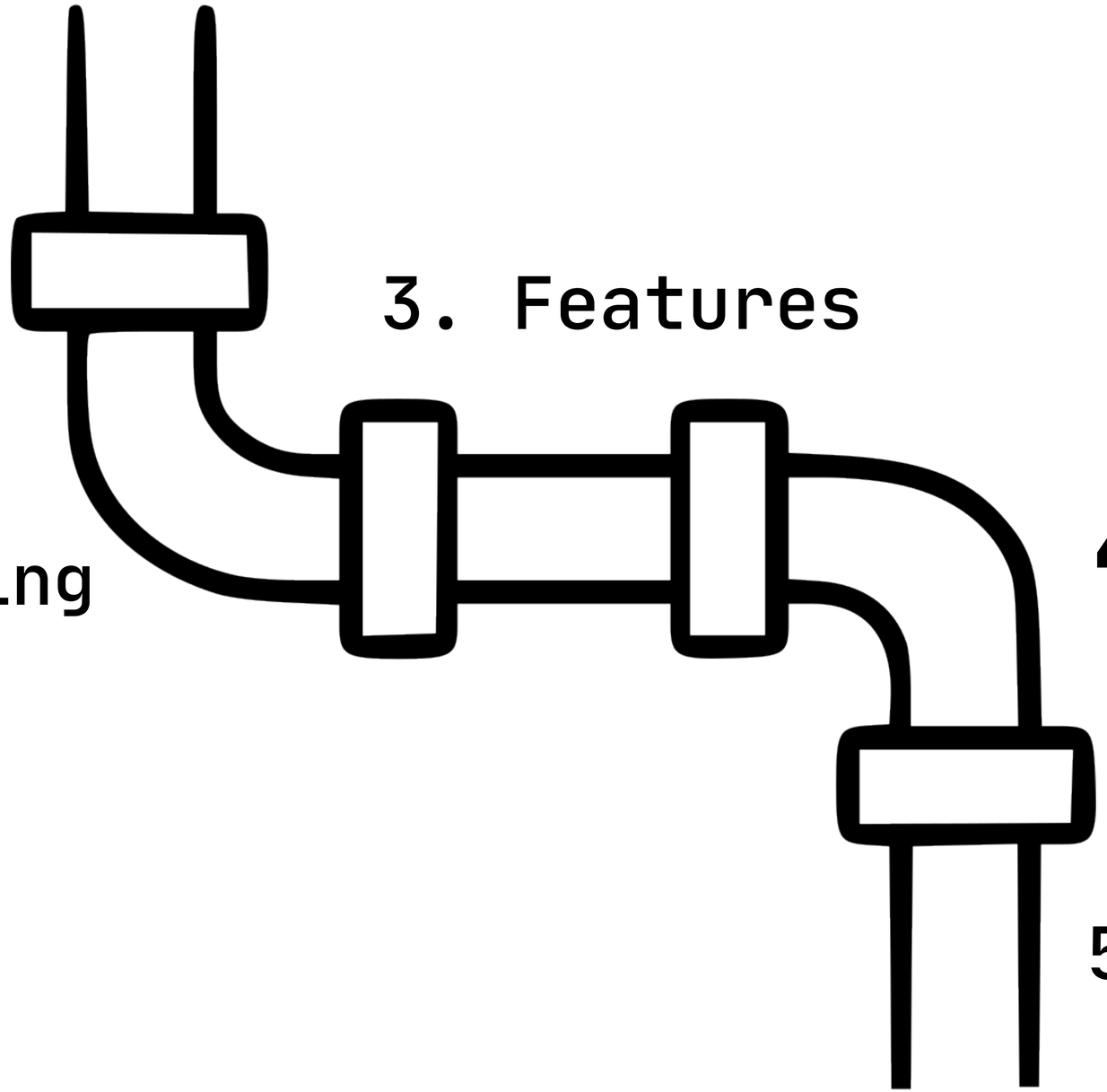
1. Setup

2. Monitoring

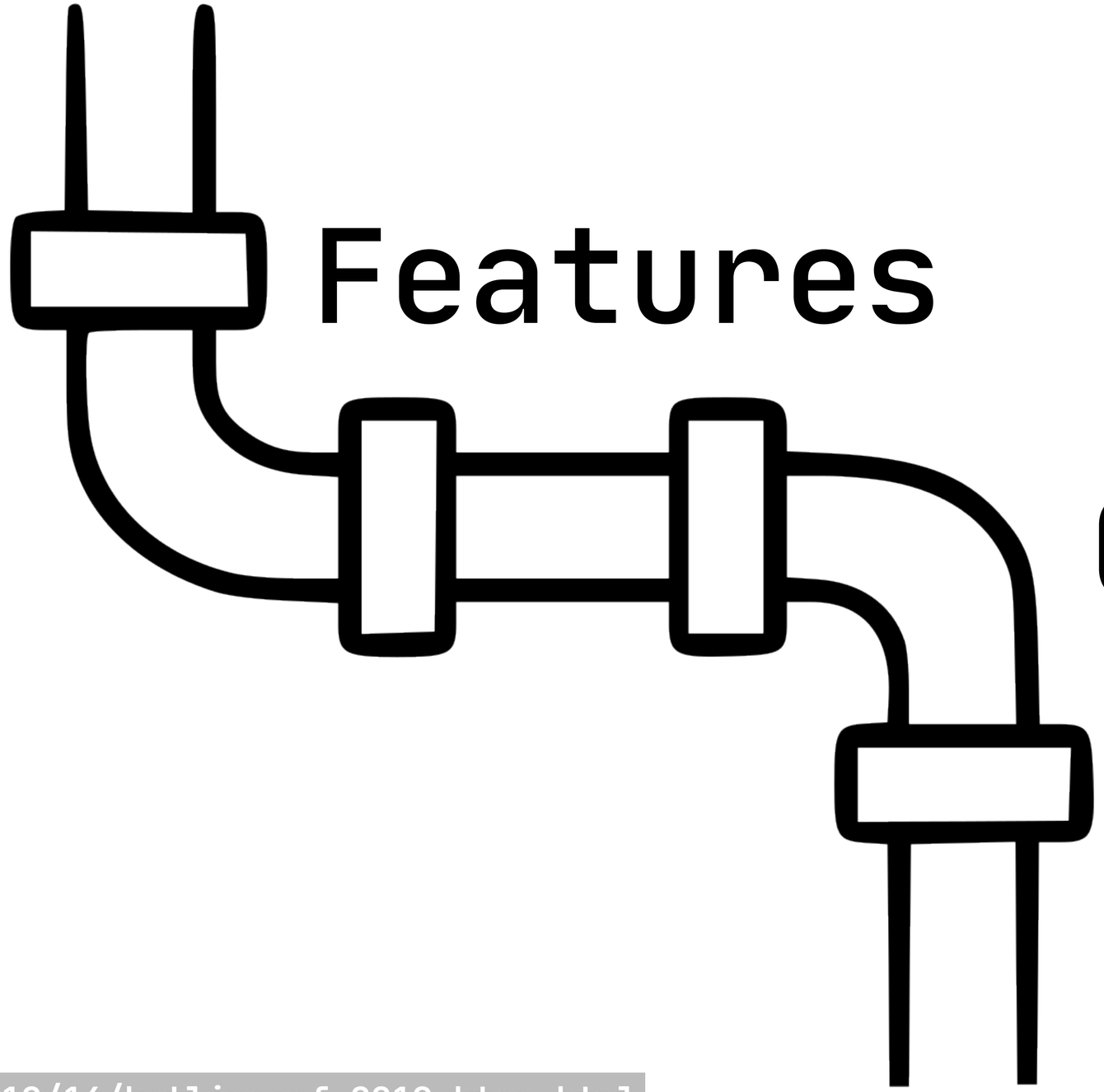
3. Features

4. Call

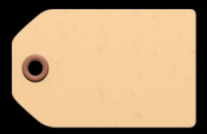
5. Fallback



https://dankim.org/2019/12/16/kotlinconf-2019-ktor.html

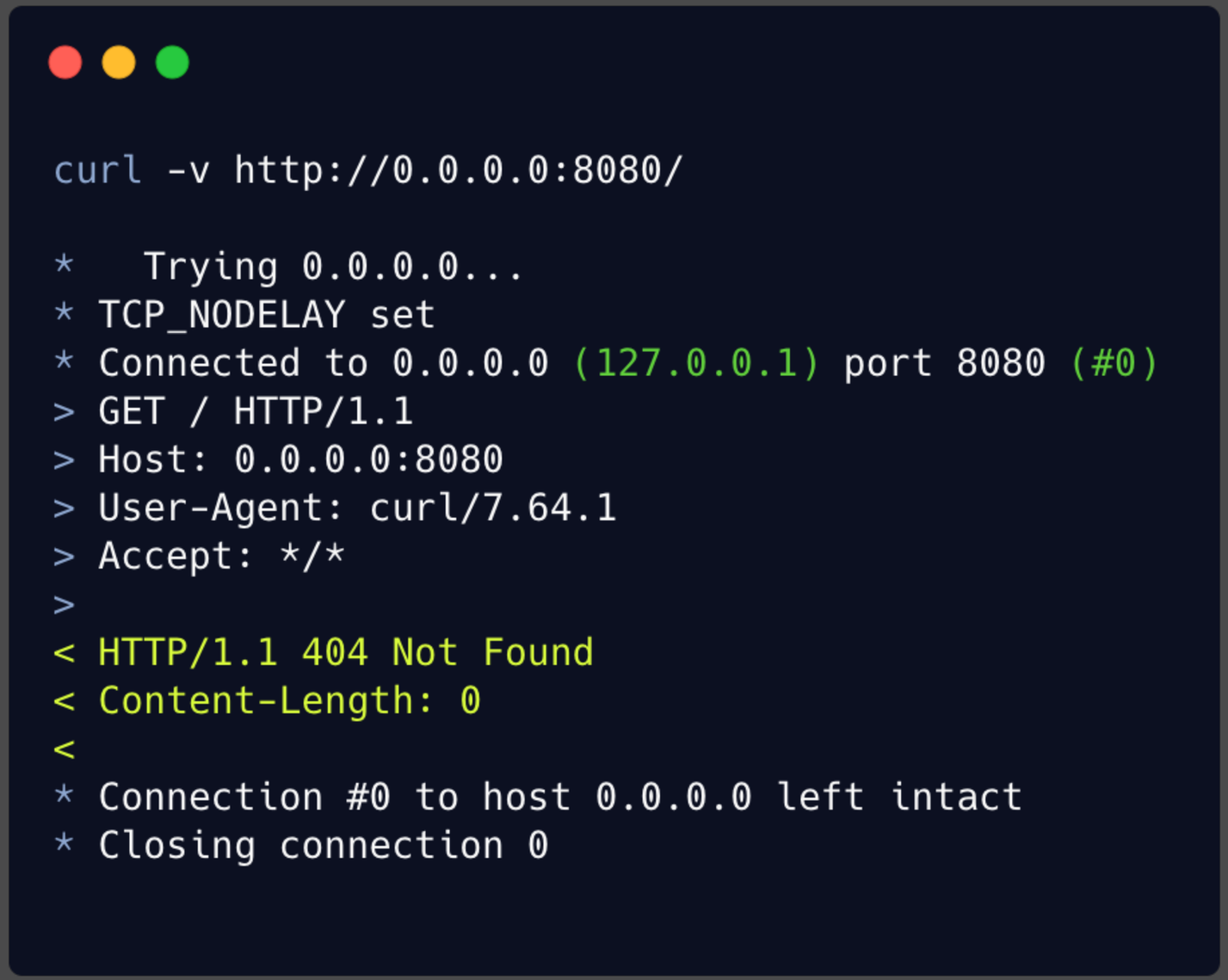


https://dankim.org/2019/12/16/kotlinconf-2019-ktor.html



Features

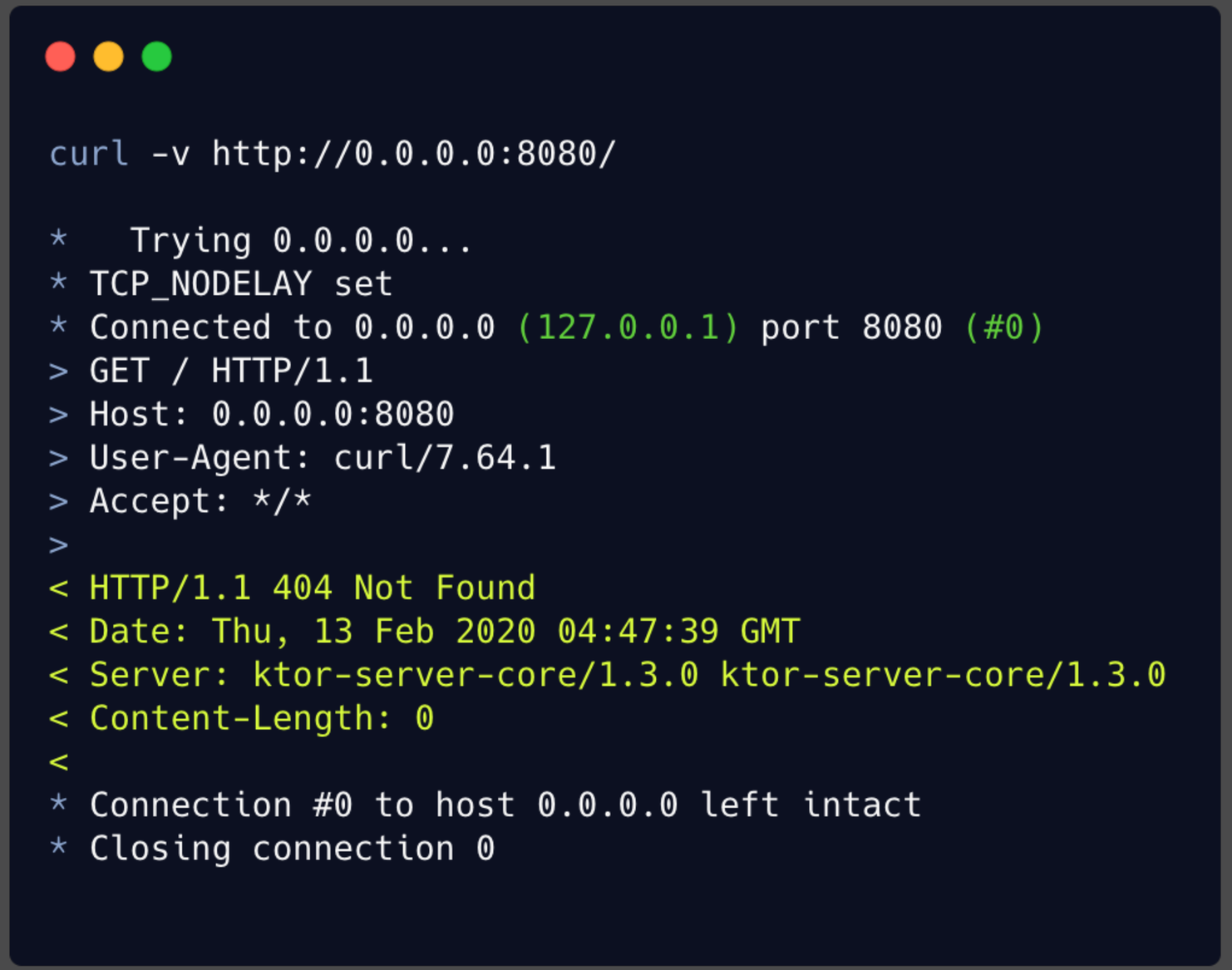
DefaultHeaders Feature


curl -v http://0.0.0.0:8080/
*   Trying 0.0.0.0...
* TCP_NODELAY set
* Connected to 0.0.0.0 (127.0.0.1) port 8080 (#0)
> GET / HTTP/1.1
> Host: 0.0.0.0:8080
> User-Agent: curl/7.64.1
> Accept: */*
>
< HTTP/1.1 404 Not Found
< Content-Length: 0
<
* Connection #0 to host 0.0.0.0 left intact
* Closing connection 0

```
fun Application.module() {
}
```

```
fun Application.module() {
    install(DefaultHeaders)
}
```

```
override fun install(...): DefaultHeaders {
    ...

    val feature = DefaultHeaders(config)
    pipeline.intercept(ApplicationCallPipeline.Features) {
      feature.intercept(call)
    }

    ...
}
```

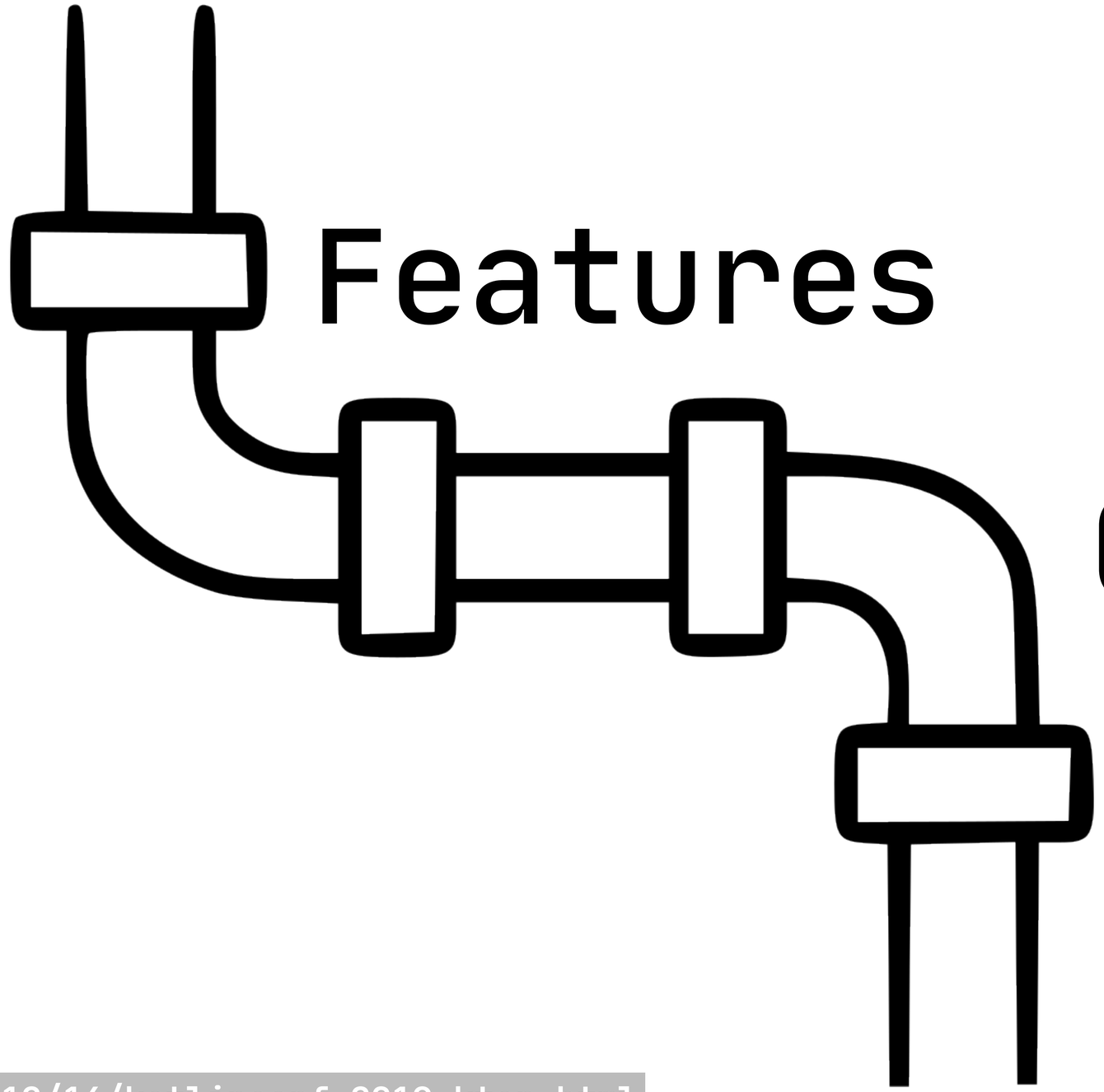

curl -v http://0.0.0.0:8080/
*   Trying 0.0.0.0...
* TCP_NODELAY set
* Connected to 0.0.0.0 (127.0.0.1) port 8080 (#0)
> GET / HTTP/1.1
> Host: 0.0.0.0:8080
> User-Agent: curl/7.64.1
> Accept: */*
>
< HTTP/1.1 404 Not Found
< Date: Thu, 13 Feb 2020 04:47:39 GMT
< Server: ktor-server-core/1.3.0 ktor-server-core/1.3.0
< Content-Length: 0
<
* Connection #0 to host 0.0.0.0 left intact
* Closing connection 0

Routes

```
fun Application.module() {
    install(DefaultHeaders)
}
```

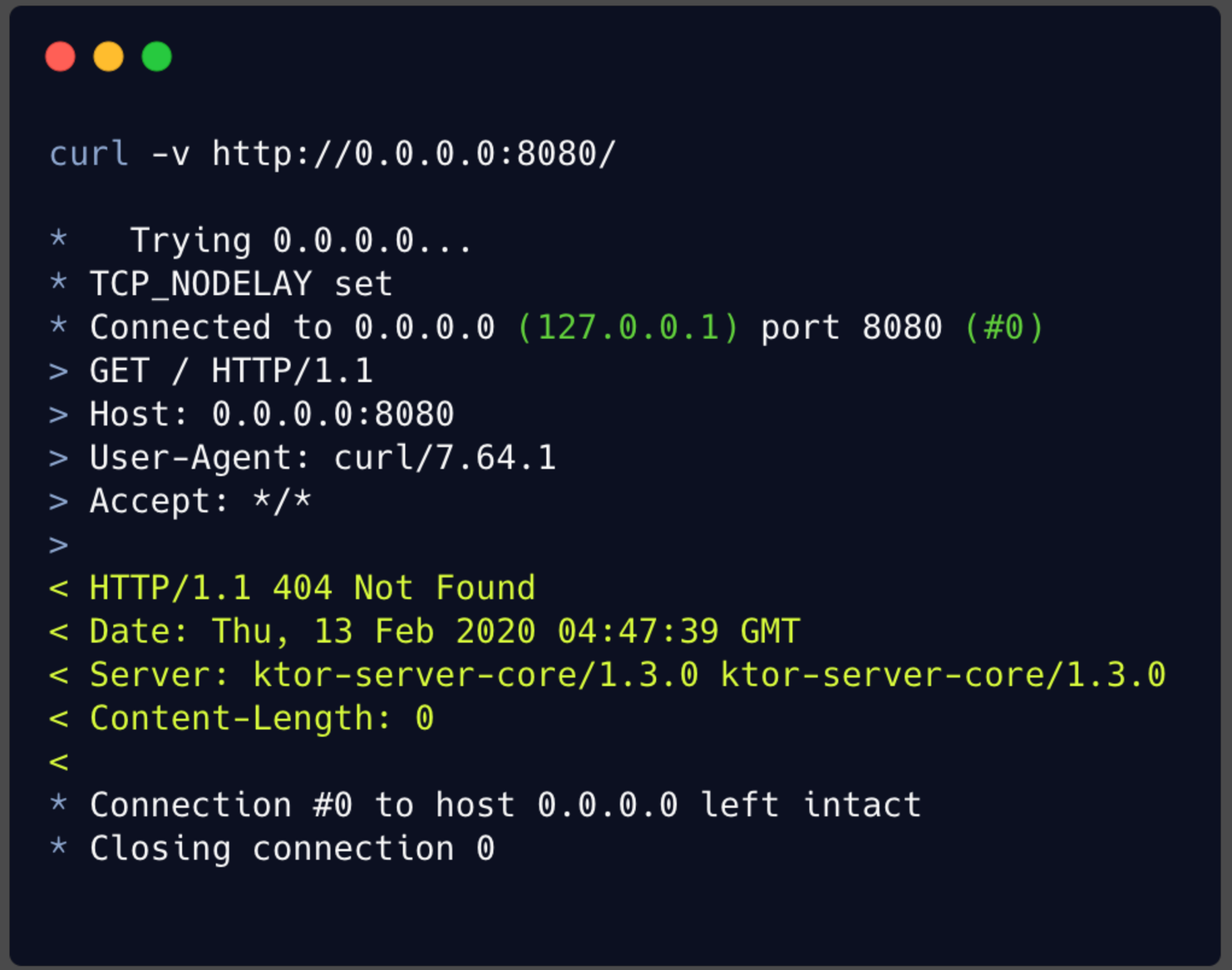
```
fun Application.module() {
    install(DefaultHeaders)

    install(Routing)
}
```

```
override fun install(...): Routing {
    ...

    val routing = Routing(pipeline).apply(configure)
    pipeline.intercept(ApplicationCallPipeline.Call) {
        routing.interceptor(this)
    }

    ...
}
```



https://dankim.org/2019/12/16/kotlinconf-2019-ktor.html

```
fun Application.module() {
    install(DefaultHeaders)

    install(Routing)
}
```


curl -v http://0.0.0.0:8080/
*   Trying 0.0.0.0...
* TCP_NODELAY set
* Connected to 0.0.0.0 (127.0.0.1) port 8080 (#0)
> GET / HTTP/1.1
> Host: 0.0.0.0:8080
> User-Agent: curl/7.64.1
> Accept: */*
>
< HTTP/1.1 404 Not Found
< Date: Thu, 13 Feb 2020 04:47:39 GMT
< Server: ktor-server-core/1.3.0 ktor-server-core/1.3.0
< Content-Length: 0
<
* Connection #0 to host 0.0.0.0 left intact
* Closing connection 0

```
fun Application.module() {
    install(DefaultHeaders)

    install(Routing)
}
```

```
fun Application.module() {
    install(DefaultHeaders)

    install(Routing) {
        get("/") {
            call.respondText {
                "Hello KotlinConf Global Medellin!”
            }
        }
    }
}
```

```
curl -v http://0.0.0.0:8080/

*   Trying 0.0.0.0...
* TCP_NODELAY set
* Connected to 0.0.0.0 (127.0.0.1) port 8080 (#0)
> GET / HTTP/1.1
> Host: 0.0.0.0:8080
> User-Agent: curl/7.64.1
> Accept: */*
>
< HTTP/1.1 200 OK
< Date: Fri, 14 Feb 2020 05:14:11 GMT
< Server: ktor-server-core/1.3.0 ktor-server-core/1.3.0
< Content-Length: 33
< Content-Type: text/plain; charset=UTF-8
<
* Connection #0 to host 0.0.0.0 left intact
Hello KotlinConf Global Medellin!
* Closing connection 0
```

Consigamos datos

GET GitHub Data

```
data class GitHubRepo(
    val id: String,
    val name: String,
    @SerializedName(name = “full_name”)
    val fullName: String,
    @SerializedName(name = “html_url")
    val htmlUrl: String,
    val description: String
)
```

```
@GET("user/repos/")
suspend fun getGitHubRepos(@Query("visibility") visibility: String):
List<GitHubRepo>
```

```
private fun getGitHubData(): List<GitHubRepo>? {
    val repos: List<GitHubRepo>? = null

    runBlocking {
        launch {
            try {
                repos = withContext(Dispatchers.IO) {
                    getGithubServicesInstance()
                        .getGitHubRepos("all")
                }
            } catch (t: Throwable) {
                t.printStackTrace()
                handleError("Ooops!, something went wrong")
            }
        }
    }

    return repos
}
```

```
private fun getGitHubData(): List<GitHubRepo>? {
    val repos: List<GitHubRepo>? = null

    runBlocking {
        launch {
            try {
                repos = withContext(Dispatchers.IO) {
                    getGithubServicesInstance()
                        .getGitHubRepos("all")
                }
            } catch (t: Throwable) {
                t.printStackTrace()
                handleError("Ooops!, something went wrong")
            }
        }
    }

    return repos
}
```

```
private fun getGitHubData(): List<GitHubRepo>? {
    val repos: List<GitHubRepo>? = null

    runBlocking {
        launch {
            try {
                repos = withContext(Dispatchers.IO) {
                    getGithubServicesInstance()
                        .getGitHubRepos("all")
                }
            } catch (t: Throwable) {
                t.printStackTrace()
                handleError("Ooops!, something went wrong")
            }
        }
    }

    return repos
}
```

```
fun Application.module() {
    install(DefaultHeaders)

    install(Routing) {

    }
}
```

```
fun Application.module() {
    install(DefaultHeaders)

    install(Routing) {
        get("/dashboard.json") {
            val gitHubData = getGitHubData()
            val gson = Gson()

            call.respondText {
              gson.toJson(gitHubData)
            }
        }
    }
}
```

```
fun Application.module() {
    install(DefaultHeaders)

    install(Routing) {
        get("/dashboard.json") {
            val gitHubData = getGitHubData()
            val gson = Gson()

            call.respondText {
              gson.toJson(gitHubData)
            }
        }
    }
}
```

```
fun Application.module() {
    install(DefaultHeaders)

    install(Routing) {
        get("/dashboard.json") {
            val gitHubData = getGitHubData()
            val gson = Gson()

                call.respondText {
                  gson.toJson(gitHubData)
                }
        }
    }
}
```

```
curl –H "Authorization: Bearer <bearer_token>"
http://0.0.0.0:8080/dashboard.json
```

```
curl -H "Authorization: Bearer <bearer_token>" http://0.0.0.0:8080/dashboard.json
```


```
[
  {
    "id":"144093733",
    "name":"adyen-android-example",
    "full_name":"alejandro-rios/adyen-android-example",
    "html_url":"https://github.com/alejandro-rios/adyen-android-example",
    "description":"Adyen example using custom UI"
  },
  {
    "id":"207712668",
    "name":"android-dependency-injection-performance",
    "full_name":"alejandro-rios/android-dependency-injection-performance",
    "html_url":"https://github.com/alejandro-rios/android-dependency-injection-performance",
    "description":"Measure the performance of several Dependency Injection frameworks in different devices"
  },
  {
    "id":"196287734",
    "name":"awesome-android-scripts",
    "full_name":"alejandro-rios/awesome-android-scripts",
    "html_url":"https://github.com/alejandro-rios/awesome-android-scripts",
    "description":"Simple scripts to make your life easier for android setups"
  },
  {
    "id":"178680612",
    "name":"Condor-Sports-Java",
    "full_name":"alejandro-rios/Condor-Sports-Java",
    "html_url":"https://github.com/alejandro-rios/Condor-Sports-Java"
  },
  {
    "id":"186741748",
    "name":"Condor-Sports-Kotlin",
    "full_name":"alejandro-rios/Condor-Sports-Kotlin",
    "html_url":"https://github.com/alejandro-rios/Condor-Sports-Kotlin"
  },
  {
    "id":"216081448",
    "name":"CopeAndroid",
    "full_name":"alejandro-rios/CopeAndroid",
    "html_url":"https://github.com/alejandro-rios/CopeAndroid"
  },
```


GitHub Repo POST

```
data class RepoParams(
    val name: String,
    val description: String,
    val private: Boolean = false
)
```

```
@POST("user/repos/")
suspend fun CreateRepo(@Body repoParams: RepoParams): GitHubRepo
```

```
post("/repos/new") {
    val postParams = call.receiveParameters()
    val repoParams = RepoParams(name = params["repoName"]!!,
                                description= params["description"]!!)

    runBlocking {
        launch {
            try {
                val response = withContext(Dispatchers.IO) {
                    getGitHubServicesInstance().createRepo(repoParams)
                }

                if (response.isSuccessful) {
                    call.respondRedirect("/dashboard")
                } else {
                    handleError(response.errorBody().toString())
                }
            } catch (t: Throwable) {
                t.printStackTrace()
                handleError("Oops!, something went wrong")
            }
        }
    }
}
```

```
post("/repos/new") {
      val postParams = call.receiveParameters()
    val repoParams = RepoParams(name = params["repoName"]!!,
                                description= params["description"]!!)

    runBlocking {
        launch {
            try {
                val response = withContext(Dispatchers.IO) {
                    getGitHubServicesInstance().createRepo(repoParams)
                }

                if (response.isSuccessful) {
                    call.respondRedirect("/dashboard")
                } else {
                    handleError(response.errorBody().toString())
                }
            } catch (t: Throwable) {
                t.printStackTrace()
                handleError("Oops!, something went wrong")
            }
        }
    }
}
```

```
post("/repos/new") {
    val postParams = call.receiveParameters()
    val repoParams = RepoParams(name = params["repoName"]!!,
                                description= params["description"]!!)


    runBlocking {
       launch {
           try {
               val response = withContext(Dispatchers.IO) {
                   getGitHubServicesInstance().createRepo(repoParams)
               }

               if (response.isSuccessful) {
                   call.respondRedirect("/dashboard")
               } else {
                   handleError(response.errorBody().toString())
               }
           } catch (t: Throwable) {
               t.printStackTrace()
               handleError("Oops!, something went wrong")
           }
       }
    }
}
```

```
post("/repos/new") {
    val postParams = call.receiveParameters()
    val repoParams = RepoParams(name = params["repoName"]!!,
                                description= params["description"]!!)

     runBlocking {
         launch {
             try {
                  val response = withContext(Dispatchers.IO) {
                      getGitHubServicesInstance().createRepo(repoParams)
                  }


                   if (response.isSuccessful) {
                    call.respondRedirect("/dashboard")
                } else {
                    handleError(response.errorBody().toString())
                }
             } catch (t: Throwable) {
                  t.printStackTrace()
                  handleError("Oops!, something went wrong")
             }
         }
     }
}
```

```
post("/repos/new") {
    val postParams = call.receiveParameters()
    val repoParams = RepoParams(name = params["repoName"]!!,
                                description= params["description"]!!)

    runBlocking {
        launch {
            try {
                 val response = withContext(Dispatchers.IO) {
                     getGitHubServicesInstance().createRepo(repoParams)
                 }

                      if (response.isSuccessful) {
                          call.respondRedirect("/dashboard")
                      } else {
                          handleError(response.errorBody().toString())
                      }
                } catch (t: Throwable) {
                 t.printStackTrace()
                 handleError("Oops!, something went wrong")
            }
        }
    }
}
```

Webapp

Authentication

OAuth 2

https://st.fbk.eu/sites/st.fbk.eu/files/styles/adaptive/public/oauth2-logo.jpg

Authorize ktor-github

ktor-github by **alejandro-rios**
wants to access your **alejandro-rios** account

Repositories
Public and private

Organization access

colombia-dev ✓

MunchkinInc ✗ Request

withinpixels ✗ Request

MedellinAndroid ✗ Grant

Authorize alejandro-rios

Authorizing will redirect to
http://githubktor.com:8080

Not owned or operated by GitHub

Created **less than a day ago**

Fewer than 10 GitHub users

Learn more about OAuth

```
val gitHubOauthProvider = OAuthServerSettings.OAuth2ServerSettings(
    name = "github",
    authorizeUrl = "https://github.com/login/oauth/authorize",
    accessTokenUrl = "https://github.com/login/oauth/access_token",
    requestMethod = HttpMethod.Post,
    clientId = OAUTH_CLIENT_ID,
    clientSecret = OAUTH_CLIENT_SECRET,
    defaultScopes = listOf("repo")
)
```

```
fun Application.module() {
    // ...

    install(Authentication) {
        oauth("github-oauth") {
            client = HttpClient(Apache)
            providerLookup = { gitHubOauthProvider }
            urlProvider = { redirectUrl("/login") }
        }
    }
}
```

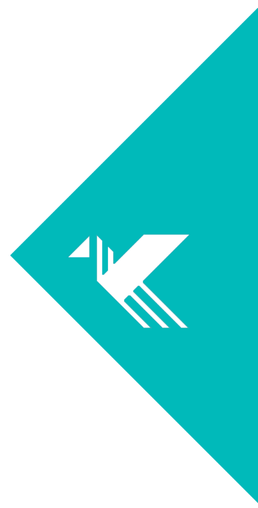
```
fun Application.module() {
    ...

    install(Routing) {
        authenticate("github-oauth") {
            route("/login") {
                handle {
                    val principal =
                      call.authentication.principal<OAuthAccessTokenResponse.OAuth2>()
                      ?: error("No principal")

                    saveOAuthToken(principal.accessToken)

                    call.respondRedirect("/")
                }
            }
        }
    }
}
```

```
fun Application.module() {
    ...

    install(Routing) {
        authenticate("github-oauth") {
            route("/login") {
                handle {
                    val principal =
                      call.authentication.principal<OAuthAccessTokenResponse.OAuth2>()
                      ?: error("No principal")

                    saveOAuthToken(principal.accessToken)

                    call.respondRedirect("/")
                }
            }
        }
    }
}
```

```
fun Application.module() {
    ...

    install(Routing) {
        authenticate("github-oauth") {
            route("/login") {
                handle {
                    val principal =
                      call.authentication.principal<OAuthAccessTokenResponse.OAuth2>()
                      ?: error("No principal")

                    saveOAuthToken(principal.accessToken)

                    call.respondRedirect("/")
                }
            }
        }
    }
}
```

Mostremos los datos

```
get("/dashboard") {
    val gitHubData = getGitHubData()

    ...
}
```

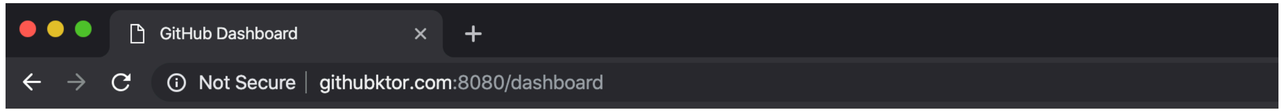


```
call.respondHtml {
  head {
      styleLink("/static-resources/styles.css")
      title { +"GitHub Dashboard:" }
  }
  body {
    imageInput {
      width = "400px"
      src = "https://i.kym-cdn.com/photos/images/original/001/704/393/8d2.png"
    }
    p {
      h1 { +"GitHub Dashboard" }
      h2 { +"Repos:" }
      h3 { a("/repos/new") { +"Create repo" } }
      ul {
          gitHubData?.forEach {
              li {
                  p {
                      +StringEscapeUtils.unescapeHtml4(it.fullname)
                      br {
                          it.description?.let { desc ->
                              +StringEscapeUtils.unescapeHtml4(desc)
                          }
                      }
                  }
              }
          }
      }
   }
}
```

```
call.respondHtml {
  head {
      styleLink("/static-resources/styles.css")
      title { +"GitHub Dashboard:" }
  }
  body {
    imageInput {
      width = "400px"
      src = "https://i.kym-cdn.com/photos/images/original/001/704/393/8d2.png"
    }
    p {
      h1 { +"GitHub Dashboard" }
      h2 { +"Repos:" }
      h3 { a("/repos/new") { +"Create repo" } }
      ul {
          gitHubData?.forEach {
              li {
                  p {
                     +StringEscapeUtils.unescapeHtml4(it.fullname)
                     br {
                         it.description?.let { desc ->
                             +StringEscapeUtils.unescapeHtml4(desc)
                         }
                     }
                  }
              }
          }
      }
   }
}
```

```
call.respondHtml {
  head {
      styleLink("/static-resources/styles.css")
      title { +"GitHub Dashboard:" }
  }
  body {
     imageInput {
       width = "400px"
       src = "https://i.kym-cdn.com/photos/images/original/001/704/393/8d2.png"
     }
     p {
      h1 { +"GitHub Dashboard" }
      h2 { +"Repos:" }
      h3 { a("/repos/new") { +"Create repo" } }
      ul {
          gitHubData?.forEach {
              li {
                  p {
                     +StringEscapeUtils.unescapeHtml4(it.fullname)
                     br {
                         it.description?.let { desc ->
                             +StringEscapeUtils.unescapeHtml4(desc)
                         }
                     }
                  }
              }
          }
      }
    }
}
```

```
call.respondHtml {
  head {
      styleLink("/static-resources/styles.css")
      title { +"GitHub Dashboard:" }
  }
   body {
     imageInput {
      width = "400px"
      src = "https://i.kym-cdn.com/photos/images/original/001/704/393/8d2.png"
    }
     p {
        h1 { +"GitHub Dashboard" }
        h2 { +"Repos:" }
        h3 { a("/repos/new") { +"Create repo" } }
        ul {
            gitHubData?.forEach {
                li {
                    p {
                        +StringEscapeUtils.unescapeHtml4(it.fullname)
                        br {
                            it.description?.let { desc ->
                                +StringEscapeUtils.unescapeHtml4(desc)
                            }
                        }
                    }
                }
            }
        }
    }
}
```



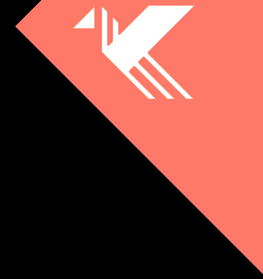
GitHub Dashboard

Repos:

Create repo

- alejandro-rios/kotlinconf-app
  KotlinConf Schedule Application
- alejandro-rios/CopeAndroid
- alejandro-rios/Hello-DI
- alejandro-rios/android-dependency-injection-performance
  Measure the performance of several Dependency Injection frameworks in different devices
- alejandro-rios/Memory-Leak-Test
  Basic sample of memory leaks detection using LeakCanary library
- alejandro-rios/GitHubKMP
  Kotlin MultiPlatform example(in progress)

Web: posting data

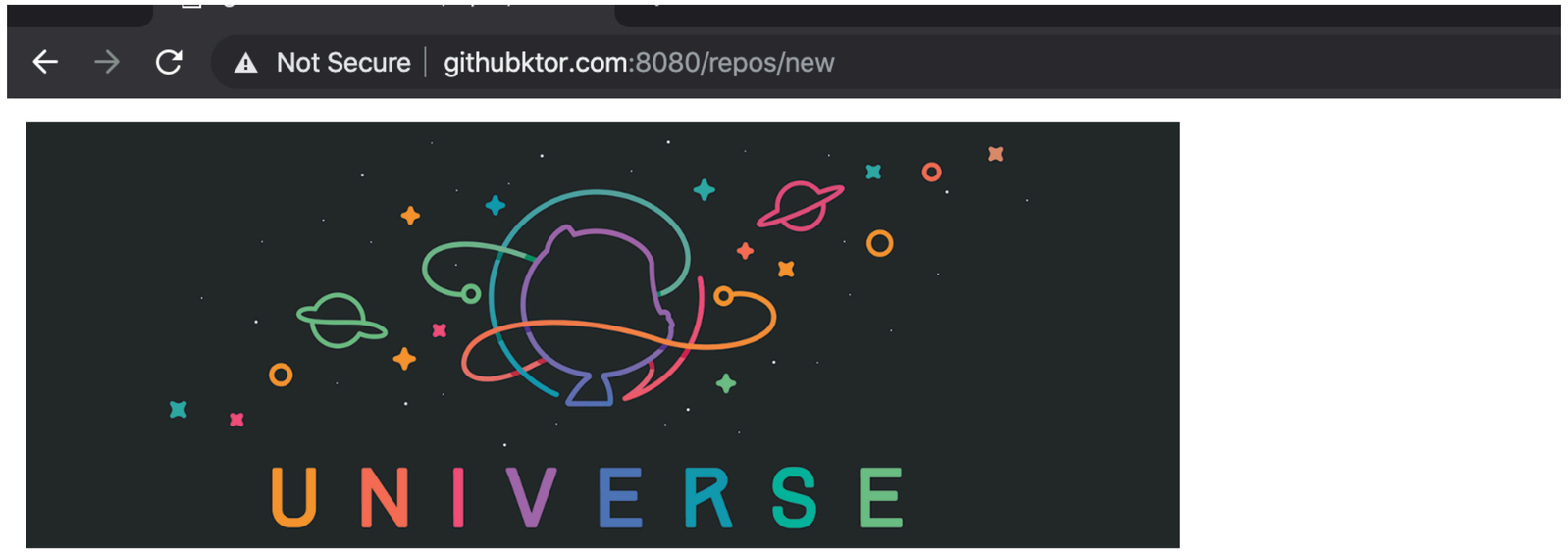


```
get("/repos/new") {
  call.respondHtml {
    head {
      styleLink("/static-resources/styles.css")
    }
    body {
      imageInput {
         width = "400px"
         src = "https://sdtimes.com/wp-content/uploads/2017/10/29682337-83f3017e-88bf-11e7-846c-138e9639b87f.png"
      }
      form("/repos/new",
        encType = FormEncType.applicationXWwwFormUrlEncoded,
        method = FormMethod.post
      ) {
        acceptCharset = "utf-8"
        p {
          label { +"Repo name: " }
          textInput { name = "repoName" }
        }
        p {
          label { +"Description: " }
          textInput { name = "description" }
        }
        p {
          submitInput { value = “create repo” }
        }
      }
    }
  }
}
```

```
get("/repos/new") {
  call.respondHtml {
    head {
      styleLink("/static-resources/styles.css")
    }
    body {
      imageInput {
         width = "400px"
         src = "https://sdtimes.com/wp-content/uploads/2017/10/29682337-83f3017e-88bf-11e7-846c-138e9639b87f.png"
      }
      form("/repos/new",
        encType = FormEncType.applicationXWwwFormUrlEncoded,
        method = FormMethod.post
      ) {
        acceptCharset = "utf-8"
        p {
          label { +"Repo name: " }
          textInput { name = "repoName" }
        }
        p {
          label { +"Description: " }
          textInput { name = "description" }
        }
        p {
          submitInput { value = “create repo” }
        }
      }
    }
  }
}
```

```
get("/repos/new") {
  call.respondHtml {
     head {
       styleLink("/static-resources/styles.css")
     }
     body {
      imageInput {
         width = "400px"
         src = "https://sdtimes.com/wp-content/uploads/2017/10/29682337-83f3017e-88bf-11e7-846c-138e9639b87f.png"
      }
      form("/repos/new",
        encType = FormEncType.applicationXWwwFormUrlEncoded,
        method = FormMethod.post
      ) {
        acceptCharset = "utf-8"
        p {
          label { +"Repo name: " }
          textInput { name = "repoName" }
        }
        p {
          label { +"Description: " }
          textInput { name = "description" }
        }
        p {
          submitInput { value = “create repo” }
        }
      }
    }
  }
}
```

```
get("/repos/new") {
  call.respondHtml {
    head {
      styleLink("/static-resources/styles.css")
    }
     body {
        imageInput {
           width = "400px"
           src = "https://sdtimes.com/wp-content/uploads/2017/10/29682337-83f3017e-88bf-11e7-846c-138e9639b87f.png"
        }
        form("/repos/new",
          encType = FormEncType.applicationXWwwFormUrlEncoded,
          method = FormMethod.post
        ) {
          acceptCharset = "utf-8"
          p {
            label { +"Repo name: " }
            textInput { name = "repoName" }
          }
          p {
            label { +"Description: " }
            textInput { name = "description" }
          }
          p {
            submitInput { value = “create repo” }
          }
        }
     }
  }
}
```



https://sdtimes.com/wp-content/uploads/2017/10/29682337-83f3017e-88bf-11e7-846c-138e9639b87f.png

GitHub Dashboard

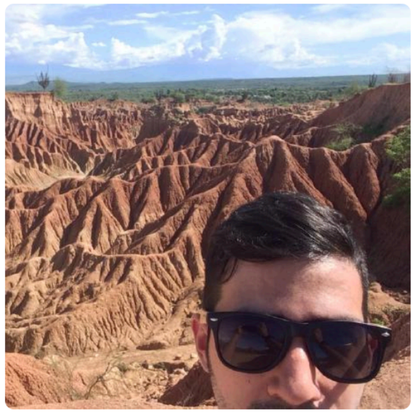
Not Secure | githubktor.com:8080/dashboard

GitHub Dashboard

Repos:

Create repo

- alejandro-rios/githubktor-test-repo
  This is an awesome test
- alejandro-rios/kotlinconf-app
  KotlinConf Schedule Application
- alejandro-rios/CopeAndroid
- alejandro-rios/Hello-DI



Alejandro Rios
alejandro-rios

Medellin, Colombia
Sign in to view email

Block or report user

Overview | **Repositories 16** | Projects 0 | Stars 26 | Followers 8 | Following 11

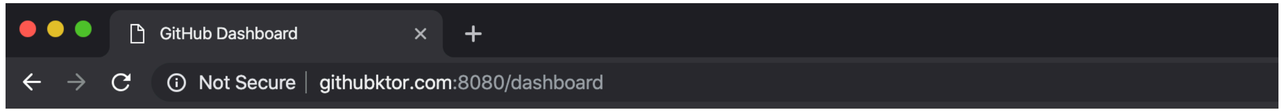
Find a repository...

githubktor-test-repo
This is an awesome test
Updated 5 minutes ago

Condor-Sports-Kotlin
Kotlin 1 Updated on Nov 25, 2019

kotlinconf-app
Forked from JetBrains/kotlinconf-app
KotlinConf Schedule Application
Kotlin 264 Apache License 2.0 Updated on Oct 31, 2019

🎨 Styling



GitHub Dashboard

Repos:

Create repo

- alejandro-rios/kotlinconf-app
  KotlinConf Schedule Application
- alejandro-rios/CopeAndroid
- alejandro-rios/Hello-DI
- alejandro-rios/android-dependency-injection-performance
  Measure the performance of several Dependency Injection frameworks in different devices
- alejandro-rios/Memory-Leak-Test
  Basic sample of memory leaks detection using LeakCanary library
- alejandro-rios/GitHubKMP
  Kotlin MultiPlatform example(in progress)

```
fun Application.module() {
    install(DefaultHeaders)

    install(Routing) { … }
}
```

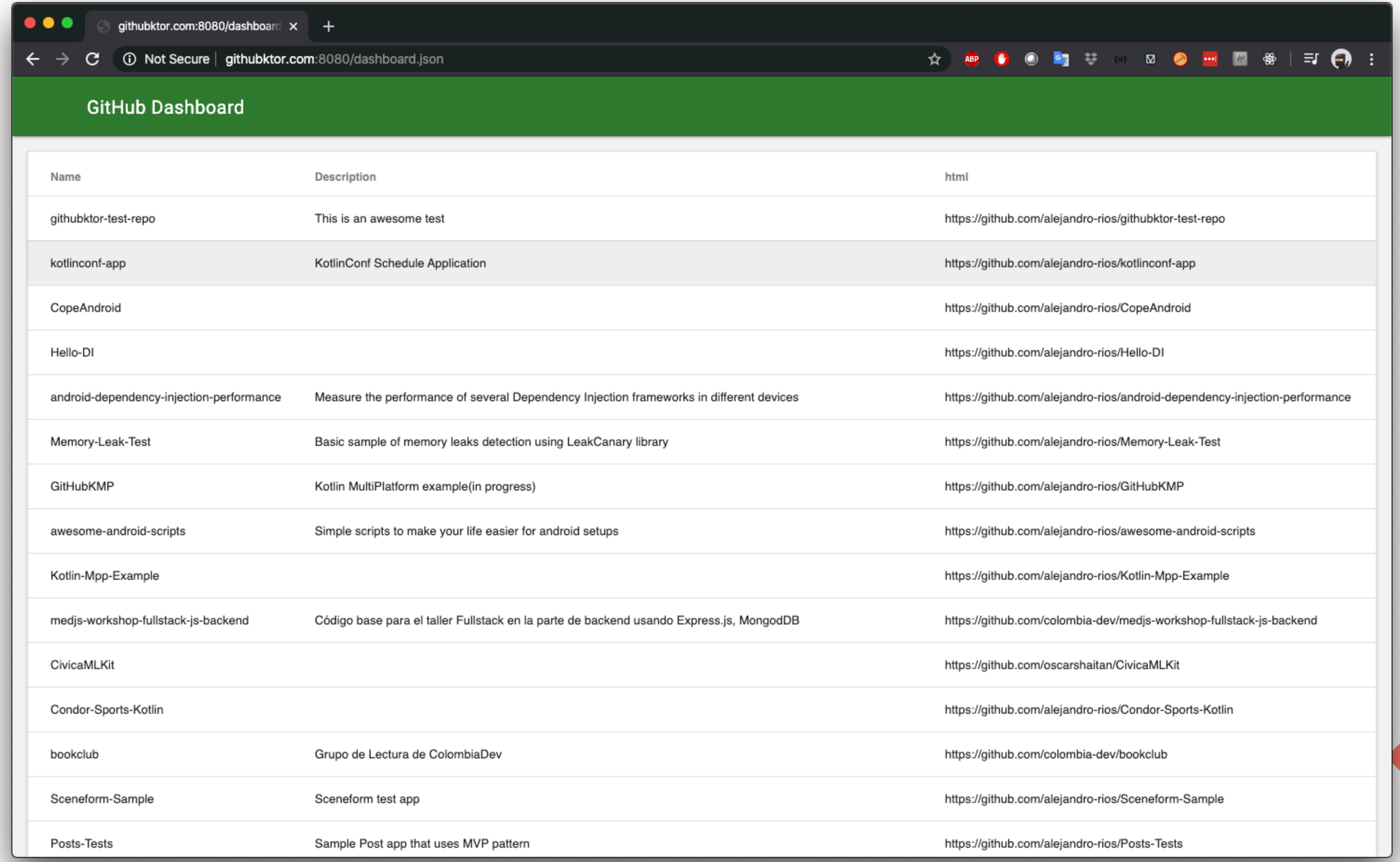
```
fun Application.module() {
    install(DefaultHeaders)

    install(Routing) { … }
     install(FreeMarker){
         acceptCharset = ClassTemplateLoader(
            this::class.java.classLoader,
            "templates"
            )
     }
}
```

```
fun Application.module() {
    install(DefaultHeaders)

     install(Routing) {
         get("/dashboard.json") {
             val gitHubData = getGitHubData()
             val gson = Gson()

             call.respondText {
               gson.toJson(gitHubData)
             }
         }
     }
    install(FreeMaker) { … }
}
```

```
fun Application.module() {
    install(DefaultHeaders)

     install(Routing) {
         get("/dashboard.json") {
             val gitHubData = getGitHubData()

             call.respond(
                 FreeMarkerContent(
                     "index.ftl",
                     mapOf(“GitHubRepos”)
                 )
             )
         }
     }
    install(FreeMaker) { … }
}
```

```
<head>
   <link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">
   <link rel="stylesheet" href="https://code.getmdl.io/1.3.0/material.indigo-pink.min.css">
   <script defer src="https://code.getmdl.io/1.3.0/material.min.js"></script>
</head>
<body>
<div class="demo-layout mdl-layout mdl-js-layout mdl-layout--fixed-header">
   <header class="demo-header mdl-layout__header mdl-color--green-800 mdl-color-text--white">
      <div class="mdl-layout__header-row">
         <span class="mdl-layout-title ">GitHub Dashboard</span>
         <div class="mdl-layout-spacer"></div>
      </div>
   </header>
   <main class="mdl-layout__content mdl-color--grey-100">
      <div class="mdl-grid">
         <table class="mdl-data-table mdl-js-data-table mdl-color--white mdl-shadow--2dp mdl-cell mdl-cell--12-col">
            <thead>
            <tr>
               <th class="mdl-data-table__cell--non-numeric">Name</th>
               <th class="mdl-data-table__cell--non-numeric">Description</th>
               <th class="mdl-data-table__cell--non-numeric">html</th>
            </tr>
            </thead>
            <tbody>
            <#list GitHubRepos as repo>
               <tr>
                  <td class="mdl-data-table__cell--non-numeric">${repo.name}</td>
                  <td class="mdl-data-table__cell--non-numeric">${repo.description!?string}</td>
                  <td class="mdl-data-table__cell--non-numeric">${repo.htmlUrl}</td>
               </tr> </#list> </tbody>
         </table>
      </div>
   </main>
</div>
```

```
<head>
  <link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">
  <link rel="stylesheet" href="https://code.getmdl.io/1.3.0/material.indigo-pink.min.css">
  <script defer src="https://code.getmdl.io/1.3.0/material.min.js"></script>
</head>
<body>
<div class="demo-layout mdl-layout mdl-js-layout mdl-layout--fixed-header">
  <header class="demo-header mdl-layout__header mdl-color--green-800 mdl-color-text--white">
    <div class="mdl-layout__header-row">
      <span class="mdl-layout-title ">GitHub Dashboard</span>
      <div class="mdl-layout-spacer"></div>
    </div>
  </header>
  <main class="mdl-layout__content mdl-color--grey-100">
    <div class="mdl-grid">
      <table class="mdl-data-table mdl-js-data-table mdl-color--white mdl-shadow--2dp mdl-cell mdl-cell--12-col">
        <thead>
        <tr>
          <th class="mdl-data-table__cell--non-numeric">Name</th>
          <th class="mdl-data-table__cell--non-numeric">Description</th>
          <th class="“mdl-data-table__cell--non-numeric">html</th>
        </tr>
        </thead>
        <tbody>
          <#list GitHubRepos as repo>
              <tr>
                <td class="mdl-data-table__cell--non-numeric">${repo.name}</td>
                <td class="mdl-data-table__cell--non-numeric">${repo.description!?string}</td>
                <td class="mdl-data-table__cell--non-numeric">${repo.htmlUrl}</td>
              </tr> </#list>
        </tbody>
      </table>
    </div>
  </main>
</div>
```



Name	Description	html
githubktor-test-repo	This is an awesome test	https://github.com/alejandro-rios/githubktor-test-repo
kotlinconf-app	KotlinConf Schedule Application	https://github.com/alejandro-rios/kotlinconf-app
CopeAndroid		https://github.com/alejandro-rios/CopeAndroid
Hello-DI		https://github.com/alejandro-rios/Hello-DI
android-dependency-injection-performance	Measure the performance of several Dependency Injection frameworks in different devices	https://github.com/alejandro-rios/android-dependency-injection-performance
Memory-Leak-Test	Basic sample of memory leaks detection using LeakCanary library	https://github.com/alejandro-rios/Memory-Leak-Test
GitHubKMP	Kotlin MultiPlatform example(in progress)	https://github.com/alejandro-rios/GitHubKMP
awesome-android-scripts	Simple scripts to make your life easier for android setups	https://github.com/alejandro-rios/awesome-android-scripts
Kotlin-Mpp-Example		https://github.com/alejandro-rios/Kotlin-Mpp-Example
medjs-workshop-fullstack-js-backend	Código base para el taller Fullstack en la parte de backend usando Express.js, MongodDB	https://github.com/colombia-dev/medjs-workshop-fullstack-js-backend
CivicaMLKit		https://github.com/oscarshaitan/CivicaMLKit
Condor-Sports-Kotlin		https://github.com/alejandro-rios/Condor-Sports-Kotlin
bookclub	Grupo de Lectura de ColombiaDev	https://github.com/colombia-dev/bookclub
Sceneform-Sample	Sceneform test app	https://github.com/alejandro-rios/Sceneform-Sample
Posts-Tests	Sample Post app that uses MVP pattern	https://github.com/alejandro-rios/Posts-Tests

Para terminar!

Bibliografia

- GitHub repo: *https://github.com/alejandro-rios/GitHubKtor*
- Ktor for Mobile developers - Fear the server no more, *https://dankim.org/2019/12/16/kotlinconf-2019-ktor.html*
- Building server backend with Ktor, *https://www.youtube.com/watch?v=V4PS3IjIzlw*
- Ktor Quick Start, *https://ktor.io/quickstart/index.html*
- Building a Rest API with Ktor, *https://medium.com/@billwixted/building-a-rest-api-with-ktor-4c322d31eb31*
- Building simple Webservices in Kotlin with Ktor, *https://medium.com/playkids-tech-blog/building-simple-webservices-in-kotlin-using-ktor-59501f004070*

https://i.ya-webdesign.com/images/server-icon-png-12.png

THANK YOU!!

Web	https://ktor.io
GitHub	ktorio / ktor
Slack	kotlinglang #ktor

Alejandro Rios