
ING

```
val kotlinTalks = kotlinMeetup("Kotlin Talks") {
    +("Kotlin, a language to rule DSLs" by "Tiberiu Tofan" from 18.30 to 19.15)
    +("Kotlin for backend" by "Cosmin Stefan" from 19.20 to 20.05)
    +("Effects in the wild, an introduction to fun…” by "Gabriel Bornea" from 20.10 to 20.55)
}
```

```
val kotlinTalks = kotlinMeetup("Kotlin Talks") {
    +"Kotlin, a language to rule DSLs" by "Tiberiu Tofan" from 18.30 to 19.15
    +"Kotlin for backend" by "Cosmin Stefan" from 19.20 to 20.05
    +"Effects in the wild, an introduction to fun…” by "Gabriel Bornea" from 20.10 to 20.55
}
```

```
infix fun String.by(author: String): Talk = Talk(this, author)
infix fun Talk.from(startTime: Double): Talk = apply { this.startTime = startTime.hours }
infix fun Talk.to(endTime: Double): Talk = apply { this.endTime = endTime.hours }
```

INFIX FUNCTIONS

```kotlin
class KotlinMeetupBuilder(private var title: String) {
    private val talks = mutableListOf<Talk>()
    //omitted code
    operator fun Talk.unaryPlus(): Talk = apply { talks.add(this) }

}
```

ɩ

)

l

ʅ

ٮ

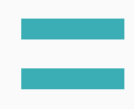
ɻ

S

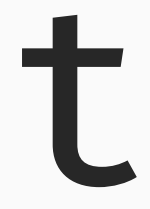

d

l

l



Ն



r

```
fun kotlinMeetup(title: String, init: KotlinMeetupBuilder.() → Unit): KotlinMeetup
fun talk(title: String, init: Talk.() → Unit): Talk
```

```
operator fun String.unaryPlus(): Talk = Talk(this).apply { talks.add(this) }
```

```
infix fun Talk.by(author: String): Talk = apply { this.author = author }
```

```
fun kotlinMeetup(title: String, init: KotlinMeetupBuilder.() → Unit): KotlinMeetup
fun talk(title: String, init: Talk.() → Unit): Talk
```

INFIX FUNCTIONS

```
infix fun String.by(author: String): Talk = Talk(this, author)
infix fun Talk.from(startTime: Double): Talk = apply { this.startTime = startTime.hours }
infix fun Talk.to(endTime: Double): Talk = apply { this.endTime = endTime.hours }
class KotlinMeetupBuilder(private var title: String) {
    private val talks = mutableListOf<Talk>()
    //omitted code
    operator fun Talk.unaryPlus(): Talk = apply { talks.add(this) }
    operator fun String.unaryPlus(): Talk = Talk(this).apply { talks.add(this) }
}
infix fun Talk.by(author: String): Talk = apply { this.author = author }
val kotlinTalks = kotlinMeetup("Kotlin Talks") {
    +"Kotlin, a language to rule DSLs" by "Tiberiu Tofan" from 18.30 to 19.15
    +"Kotlin for backend" by "Cosmin Stefan" from 19.20 to 20.05
    +"Effects in the wild, an introduction to fun…” by "Gabriel Bornea" from 20.10 to 20.55
}
```

```
fun kotlinMeetup(title: String, init: KotlinMeetupBuilder.() → Unit): KotlinMeetup
fun talk(title: String, init: Talk.() → Unit): Talk
```

INFIX FUNCTIONS

```
infix fun String.by(author: String): Talk = Talk(this, author)
infix fun Talk.from(startTime: Double): Talk = apply { this.startTime = startTime.hours }
infix fun Talk.to(endTime: Double): Talk = apply { this.endTime = endTime.hours }
class KotlinMeetupBuilder(private var title: String) {
    private val talks = mutableListOf<Talk>()
    //omitted code
    operator fun Talk.unaryPlus(): Talk = apply { talks.add(this) }
    operator fun String.unaryPlus(): Talk = Talk(this).apply { talks.add(this) }
}
infix fun Talk.by(author: String): Talk = apply { this.author = author }


    +"Kotlin, a language to rule DSLs" by "Tiberiu Tofan" from 18.30 to 19.15
    +"Kotlin for backend" by "Cosmin Stefan" from 19.20 to 20.05
    +"Effects in the wild, an introduction to fun…" by "Gabriel Bornea" from 20.10 to 20.55
}
```

ING