[Daily]
Kotlin tooling QA

Liliia Abdulina
JetBrains, Kotlin team

Kotlin tooling QA

- What are you doing?
- How do you do that?
- How do they getting into it?
- Fun?

What are you doing?

- JetBrains, Kotlin
- Tooling
- Environment
- Open tracker
- Documentation

What are you doing?

- JetBrains, **Kotlin**
- Tooling
- Environment
- Open tracker
- Documentation

Kotlin

- JVM, JS, Native (Windows, Linux, MacOS, iOS)
- Scripts & Kotlin DSL
- Mobile, Desktop, Web, Server, Embedded…
- Uber, Pinterest, Coursera…

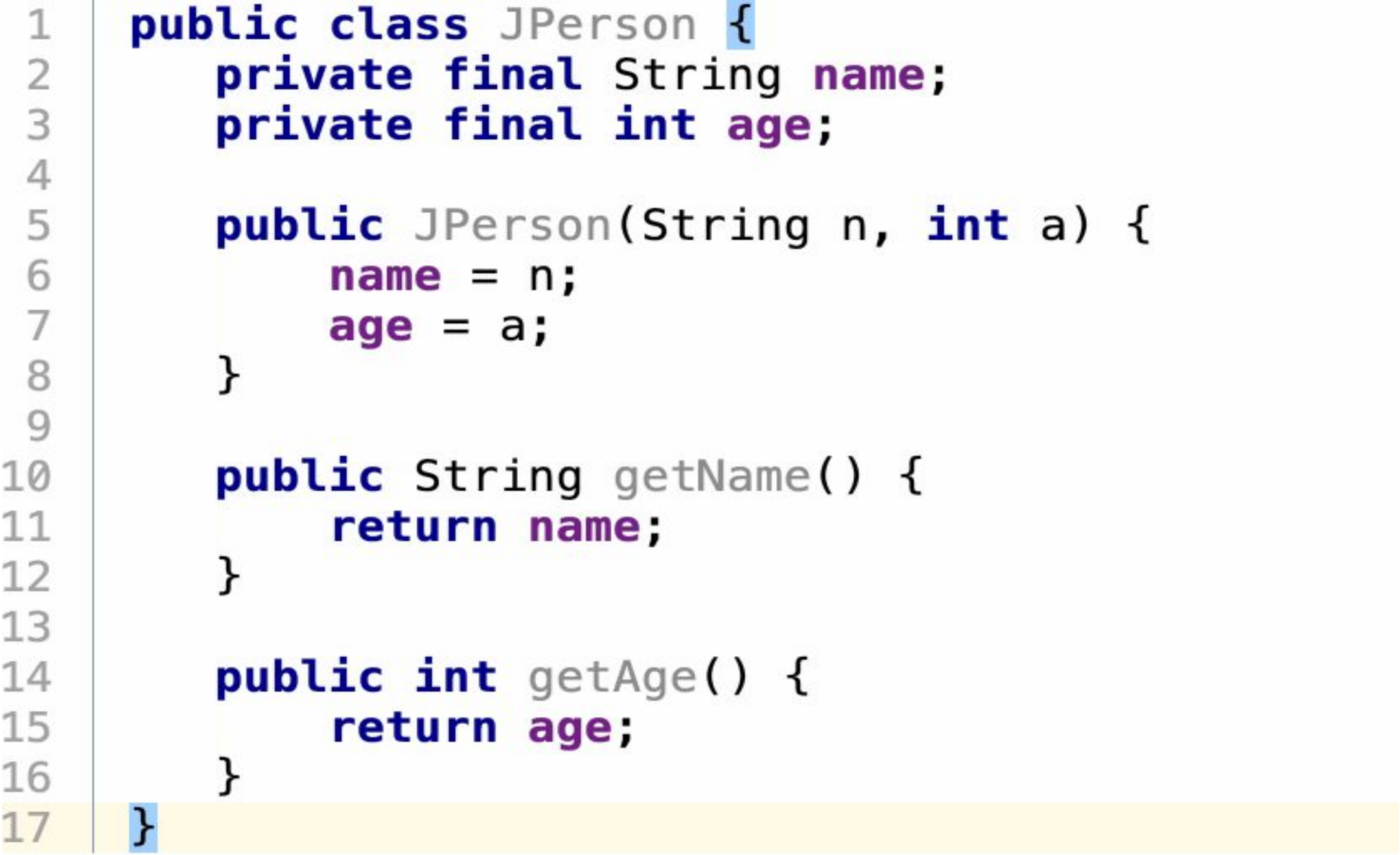
Kotlin is concise

```java
public class JPerson {
   private final String name;
   private final int age;

   public JPerson(String n, int a) {
       name = n;
       age = a;
   }

   public String getName() {
       return name;
   }

   public int getAge() {
       return age;
   }
}                            // Java
```

Kotlin is concise

```
public class JPerson {
   private final String name;
   private final int age;

   public JPerson(String n, int a) {
       name = n;
       age = a;
   }

   public String getName() {
       return name;
   }

   public int getAge() {
       return age;
   }
}                            // Java
```

```
class JPerson(val name: String,
    val age: Int)

                          // Kotlin
```



```java
public class JPerson {
    private final String name;
    private final int age;

    public JPerson(String n, int a) {
        name = n;
        age = a;
    }

    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }
}
```

Feature types

- IDE features
    - Refactorings, inspections, dialogs, ...

IDE features: refactoring

```kotlin
fun f() {
    val a = 1
    val n = a + 1
    val b = n + 1
}
```

IDE features: refactoring

```
fun f() {
    val a = 1
    val n = a + 1
    val b = n + 1
}
```

```
// Refactor > Extract > Function
```

IDE features: refactoring

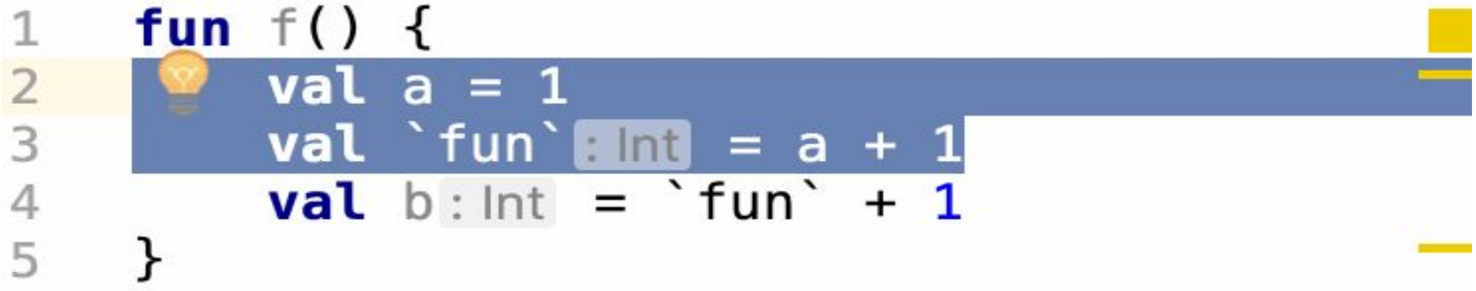
```
fun f() {
    val a = 1
    val n = a + 1
    val b = n + 1
}
```

```
fun f() {
    val n = i()
    val b = n + 1
}

private fun i(): Int {
    val a = 1
    val n = a + 1
    return n
}
```

IDE features: refactoring

```
fun f() {
    val a = 1
    val `fun` = a + 1
    val b = `fun` + 1
}
```

```
// Refactor > Extract > Function
```

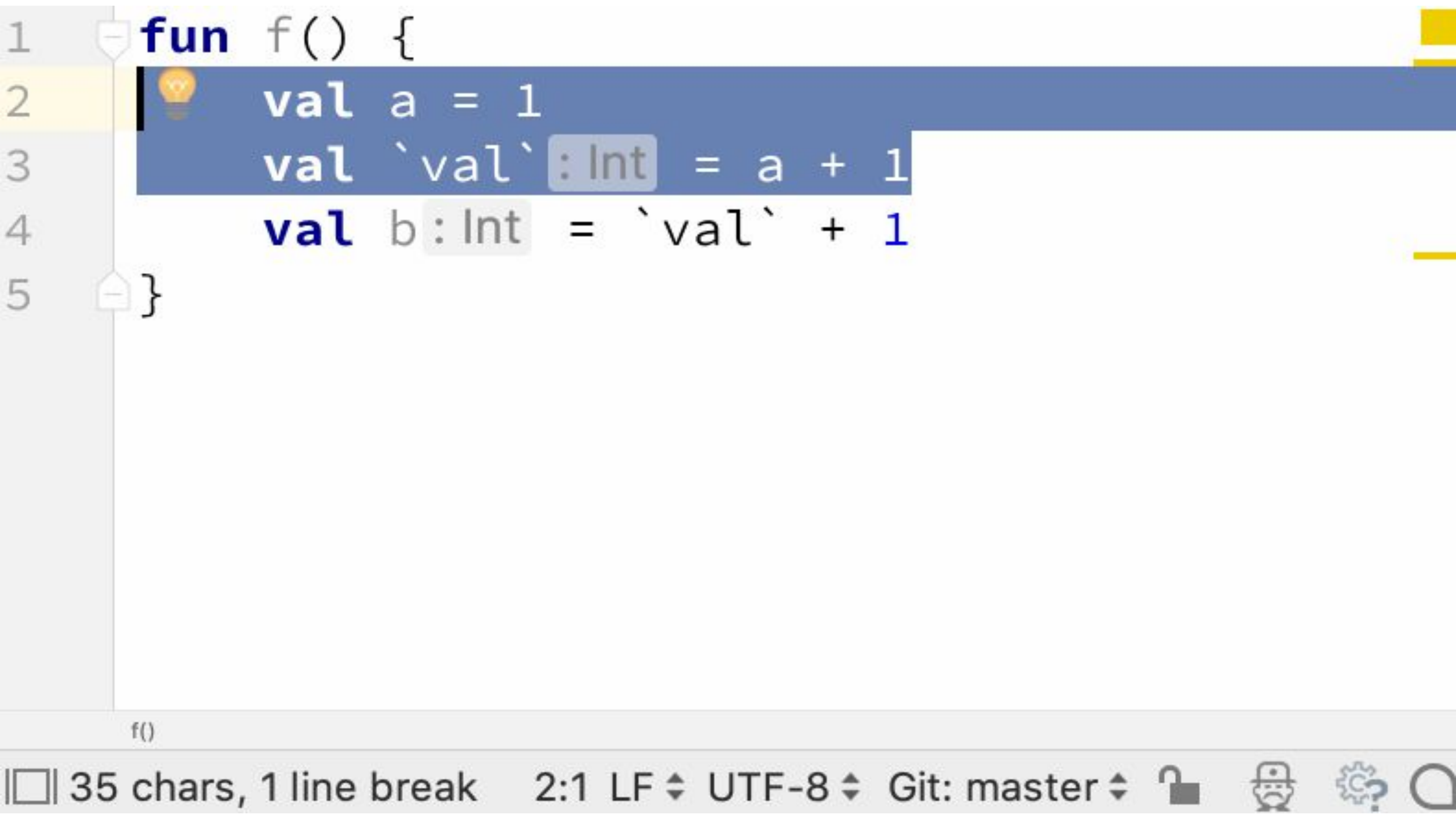


```
fun f() {
    val a = 1
    val `fun`: Int = a + 1
    val b: Int = `fun` + 1
}
```

IDE features: refactoring

```
fun f() {
    val a = 1
    val `val` = a + 1
    val b = `val` + 1
}
```

```
// Refactor > Extract > Function
```



```kotlin
fun f() {
    val a = 1
    val `val`: Int = a + 1
    val b: Int = `val` + 1
}
```

Let’s go deeper

```kotlin
val strings = listOf("Orange", "Apple", "Carrot")
```

Let’s go deeper

```kotlin
val strings = listOf("Orange", "Apple", "Carrot")

strings.filter { it.length == 6 }
```

Lambdas

```
val strings = listOf("Orange", "Apple", "Carrot")

strings.filter { it.length == 6 }
               // lambda; it: String
```

Lambdas

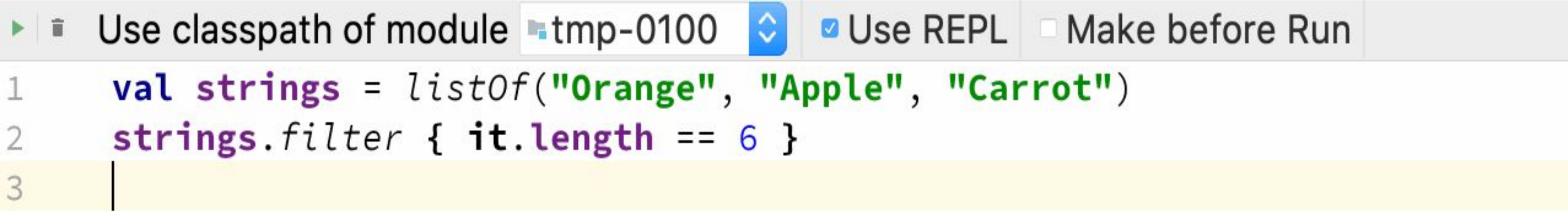
```
/**
* Returns a list containing only elements matching the given
[predicate].
*/
public inline fun <T> Iterable<T>
       .filter(predicate: (T) -> Boolean): List<T> {
   /* ... */
}
```

IDE features: scratch files

```
val strings = listOf("Orange", "Apple", "Carrot")

strings.filter { it.length == 6 }
```

IDE features: scratch files

```kotlin
val strings = listOf("Orange", "Apple", "Carrot")

strings.filter { it.length == 6 }
```



Use classpath of module tmp-0100 ☑ Use REPL □ Make before Run

```kotlin
val strings = listOf("Orange", "Apple", "Carrot")
strings.filter { it.length == 6 }

```

IDE features: intentions

- Conversion: `run` ↔ `let`, `apply` ↔ `also`
    - Sometimes it is useful to replace a standard scoping function accepting a lambda ***with receiver*** with a standard scoping function accepting a lambda ***with single parameter*** (e.g., to avoid names clash, or to simplify some sub-expression).

`run` with receiver

```kotlin
/**
* Calls the specified function [block] with `this` value as
its receiver and returns its result.
*/
@kotlin.internal.InlineOnly
public inline fun <T, R> T.run(block: T.() -> R): R {
    /* ... */
}
```

`let` with parameter

```kotlin
/**
* Calls the specified function [block] with `this` value as
its argument and returns its result.
*/
@kotlin.internal.InlineOnly
public inline fun <T, R> T.let(block: (T) -> R): R {
    /* ... */
}
```

IDE features: intentions

```
open class Person(val firstName: String)
class Employee(name: String, var manager: Person?) : Person(name)

val employee = Employee("e", Person("m"))

val test = employee.also {
   it.manager?.run { // replace ‘run’ with ‘let’
       println("${it.firstName} has a manager")
   }
}
```

IDE features: intentions

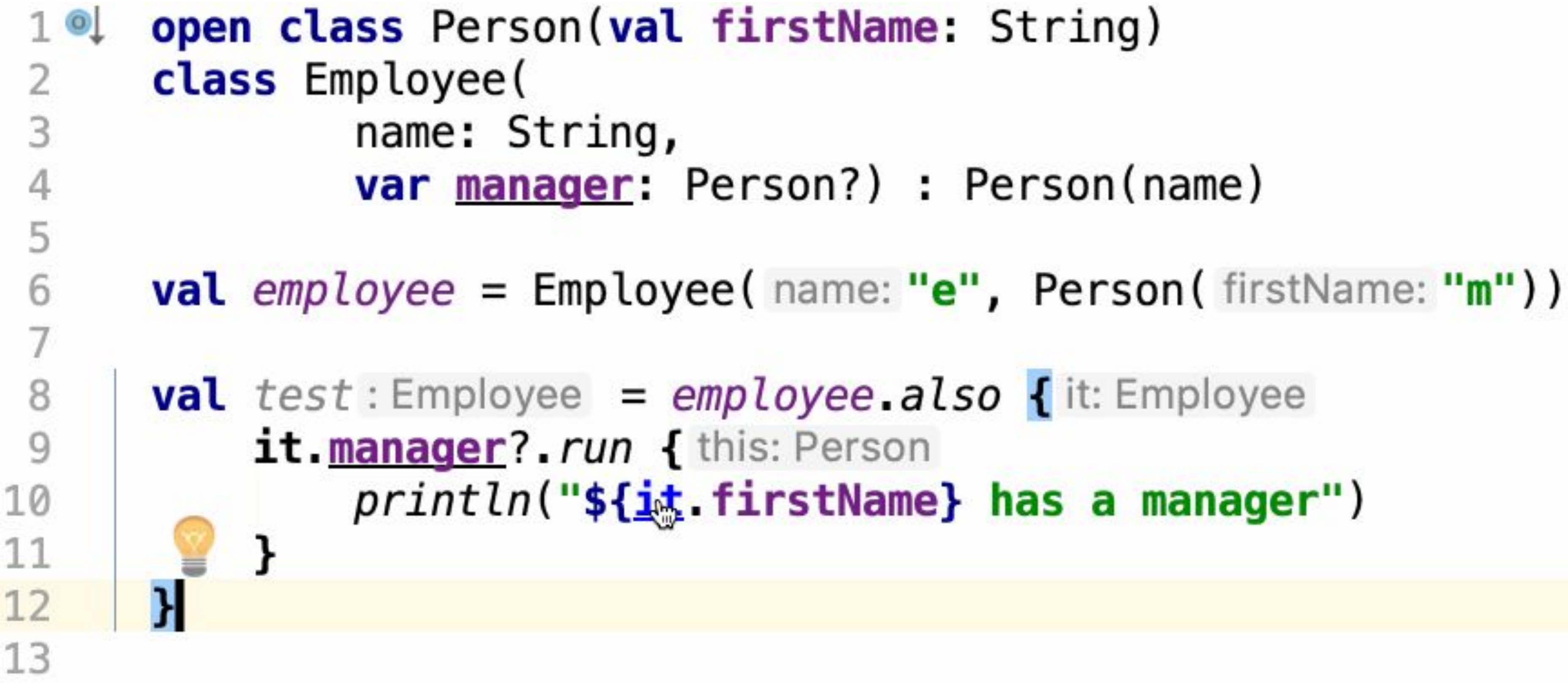
```
open class Person(val firstName: String)
class Employee(name: String, var manager: Person?) : Person(name)

val employee = Employee("e", Person("m"))

val test = employee.also {
   it.manager?.run { // replace ‘run’ with ‘let’
       println("${it.firstName} has a manager")
   }
}
```

IDE features: intentions

```
open class Person(val firstName: String)
class Employee(name: String, var manager: Person?) : Person(name)

val employee = Employee("e", Person("m"))

val test = employee.also {
   it.manager?.run { // replace ‘run’ with ‘let’
       println("${it.firstName} has a manager")
   }
}
```

IDE features: intentions

```
open class Person(val firstName: String)
class Employee(name: String, var manager: Person?) : Person(name)

val employee = Employee("e", Person("m"))

val test = employee.also {
   it.manager?.run { // replace ‘run’ with ‘let’
       println("${it.firstName} has a manager")
   }
}
```



```
open class Person(val firstName: String)
class Employee(
        name: String,
        var manager: Person?) : Person(name)

val employee = Employee( name: "e", Person( firstName: "m"))

val test : Employee  = employee.also { it: Employee
    it.manager?.run { this: Person
        println("${it.firstName} has a manager")
    }
}
```

Feature types

- IDE features
  - Refactorings, inspections, dialogs, ...
- Language features
  - Multiplatform, annotations support, compiler flags, …

Kotlin tooling includes but not limited to

- New file, module & project wizard
- Autocomplete, highlighting, code style
- Refactorings, inspections, intentions
- Run configurations, builders, debug

Who is there?

- 63 team members
  - *+ contributors*
- 1 compiler QA
- 5 tooling QA
  - *3 tooling QA before the August :)*

What are you doing?

- JetBrains, Kotlin
- Tooling
- **Environment**
- Open tracker
- Documentation

Test environment

- Kotlin plugin versions (2 or 3)

Test environment

- Kotlin plugin versions (2 or 3)
- IDEAs and Studios: (2 or 3) + (2 or 3)

Test environment

- Kotlin plugin versions (2 or 3)
- IDEAs and Studios: (2 or 3) + (2 or 3)
- Gradle versions (many :))

Test environment

- Kotlin plugin versions (2 or 3)
- IDEAs and Studios: (2 or 3) + (2 or 3)
- Gradle versions (many :))
- Libraries (many :))
- … (even more)

What are you doing?

- JetBrains, Kotlin
- Tooling
- Environment
- **Open tracker**
- Documentation

Open tracker

- kotl.in: https://youtrack.jetbrains.com/issues/KT

Open tracker

- kotl.in: https://youtrack.jetbrains.com/issues/KT
- Users

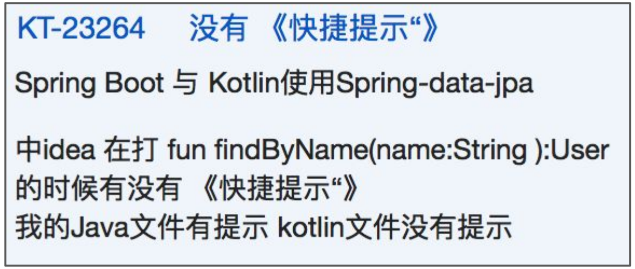
Open tracker

- kotl.in: https://youtrack.jetbrains.com/issues/KT
- Users
- Contributors

Open tracker

- kotl.in: https://youtrack.jetbrains.com/issues/KT
- Users
- Contributors
- Enthusiasts

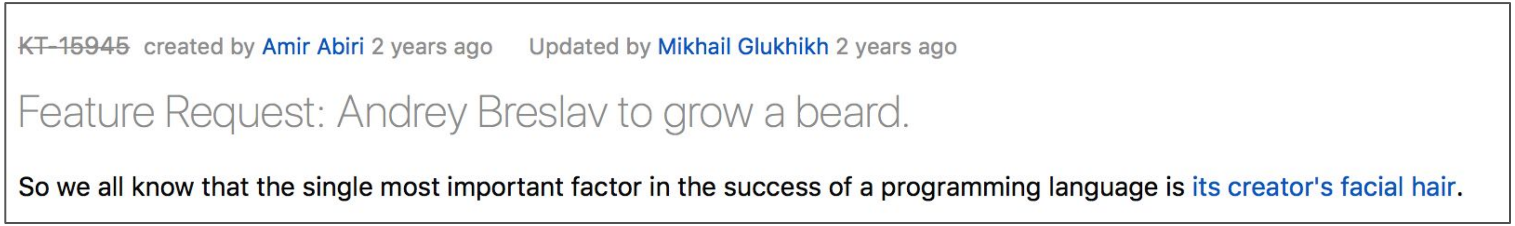
Open tracker: tickets



KT-23264　没有 《快捷提示“》

Spring Boot 与 Kotlin使用Spring-data-jpa

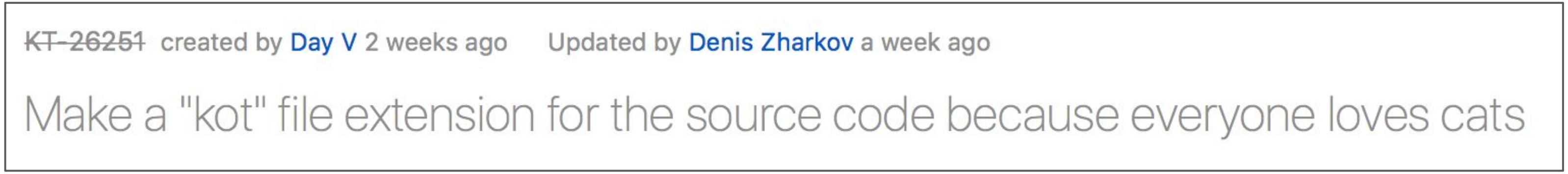
中idea 在打 fun findByName(name:String):User
的时候有没有 《快捷提示“》
我的Java文件有提示 kotlin文件没有提示



~~KT-15945~~ created by Amir Abiri 2 years ago　Updated by Mikhail Glukhikh 2 years ago

Feature Request: Andrey Breslav to grow a beard.

So we all know that the single most important factor in the success of a programming language is its creator's facial hair.



~~KT-26251~~ created by Day V 2 weeks ago　Updated by Denis Zharkov a week ago

Make a "kot" file extension for the source code because everyone loves cats

Open tracker: stories

- Google the code

Open tracker: stories

- Google the code
- Code screenshots & screencasts

Open tracker: stories

- Google the code
- Code screenshots & screencasts
- Projects with thousands of code lines

What are you doing?

- JetBrains, Kotlin
- Tooling
- Environment
- Open tracker
- **Documentation**

Documentation

- YouTrack issues

Documentation

- YouTrack issues
- Developers and community

Documentation

- YouTrack issues
- Developers and community
- Design meeting notes & KEEP

Documentation

- YouTrack issues
- Developers and community
- Design meeting notes & KEEP
- Git commit messages & code

Documentation

- YouTrack issues
- Developers and community
- Design meeting notes & KEEP
- Git commit messages & code
- Slack

What are you doing?

- JetBrains, Kotlin
- Tooling
- Environment
- Open tracker
- **Documentation**

How do you do that?

How do you do that?

- Prioritise

How do you do that?

- Prioritise
- Explore

How do you do that?

- Prioritise
- Explore
- Systematize

How do you do that?

- Prioritise
- Explore
- Systematize
- Teamwork in terms of workload

How do you do that?

- Prioritise
- Explore
- Systematize
- Teamwork in terms of workload
- Share some domains & knowledge

How do you do that?

- ⇪ Prioritise
- 🔍 Explore
- ∑ Systematize
- ⚖ Teamwork in terms of workload
- ↪ Share some domains & knowledge

How do you do that?

- ⇧ Prioritise
- 🔍 Explore
- ∑ Systematize
- ⚖ Teamwork in terms of workload
- ↪ Share some domains & knowledge

⇧ Prior
🔍 Expl
∑ Syst
⚖ Team
↪ Share

What you do again?

- Release testing
- New features testing
- Reproduce & reduce
- Fixes verification
- Infrastructure tasks
- …

What you do again?

- **Release testing**
- New features testing
- Reproduce & reduce
- Fixes verification
- Infrastructure tasks
- …

Release testing

⇪ Prior
🔍 Expl
∑ Syst
⚖️ Team
↪ Share

Release testing

⇪ The highest priority

⇪	Prior
🔍	Expl
Σ	Syst
⚖️	Team
↪	Share

Release testing

- ⇧ The highest priority
- 🔍 The most formalized flow

⇧	Prior
🔍	Expl
Σ	Syst
⚖	Team
↪	Share

Release testing

- ⇧ The highest priority
- 🔍 The most formalized flow
- ∑ Test projects, cases & tracking doc

⇧ Prior
🔍 Expl
∑ Syst
⚖️ Team
↪ Share

Release testing

- ⇪ The highest priority
- 🔍 The most formalized flow
- ∑ Test projects, cases & tracking doc
- ⚖ Release manager & dogfooding

⇪ Prior
🔍 Expl
∑ Syst
⚖ Team
↪ Share

Release testing

- ⇧ The highest priority
- 🔍 The most formalized flow
- ∑ Test projects, cases & tracking doc
- ⚖ Release manager & dogfooding
- ↪ Acceptance & urgent activities

⇧	Prior
🔍	Expl
∑	Syst
⚖	Team
↪	Share

What you do again?

- Release testing
- **Reproduce & reduce**
- New features testing
- Fixes verification
- Infrastructure tasks
- …

Reproduce & reduce

- ⇧ Prior
- 🔍 Expl
- ∑ Syst
- ⚖️ Team
- ↪ Share

Reproduce & reduce

- ⇪ Lower than release, sporadically

⇪ Prior
🔍 Expl
∑ Syst
⚖ Team
↪ Share

Reproduce & reduce

- ⇧ Lower than release, sporadically
- 🔍 Reproduce with initial STR & Reduce
  - ○ Get the details: build, IDE version, additional plugins…

⇧ Prior
🔍 Expl
∑ Syst
⚖ Team
↪ Share

Reproduce & reduce

- ⇧ Lower than release, sporadically
- 🔍 Reproduce with initial STR & Reduce
    - ○ Get the details: build, IDE version, additional plugins…
- ∑ Describe the minimal STR

⇧	Prior
🔍	Expl
∑	Syst
⚖️	Team
↪	Share

Reproduce & reduce

- ⇧ Prior
- 🔍 Expl
- ∑ Syst
- ⚖ Team
- ↪ Share

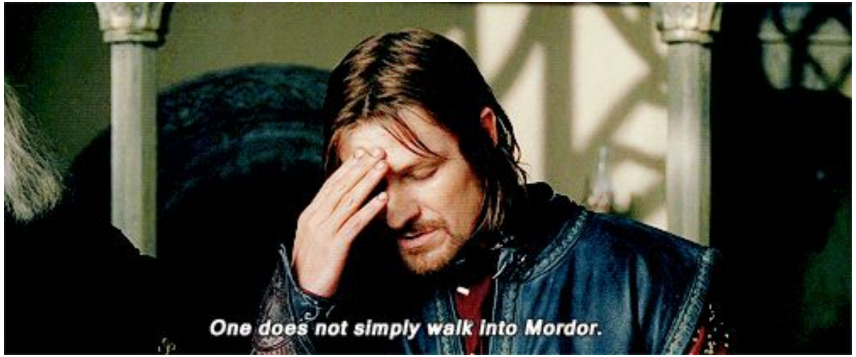
⇧ Lower than release, sporadically

🔍 Reproduce with initial STR & Reduce

- ○ Get the details: build, IDE version, additional plugins…

∑ Describe the minimal STR

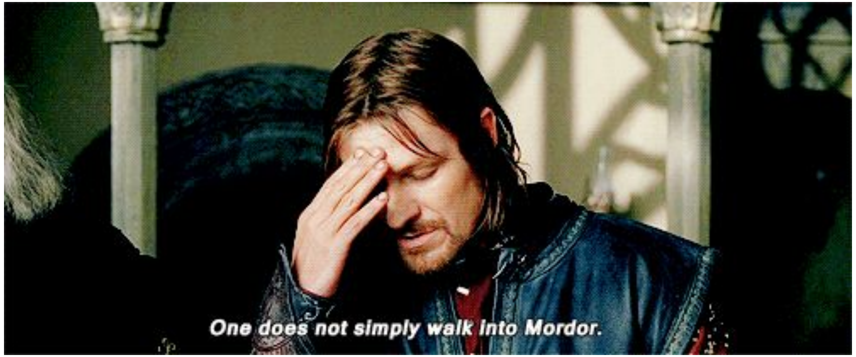
⚖ Ask the developers



Reproduce & reduce

- ⇧ Lower than release, sporadically
- 🔍 Reproduce with initial STR & Reduce
  - ○ Get the details: build, IDE version, additional plugins…
- ∑ Describe the minimal STR
- ⚖ Ask the developers
- ↪ Share the case

⇧	Prior
🔍	Expl
∑	Syst
⚖	Team
↪	Share



What you do again?

- Release testing
- Reproduce & reduce
- **New features testing**
- Fixes verification
- Infrastructure tasks
- ...

New features testing

⇪ Prior
🔍 Expl
∑ Syst
⚖️ Team
↪ Share

New features testing

- ⇧ “Big” & “small” features

⇧ Prior
🔍 Expl
∑ Syst
⚖️ Team
↪ Share

New features testing

- ⇪ “Big” & “small” features
- 🔍 Domain knowledge > oracle
  - *Several obvious bugs as a side effect*

⇪	Prior
🔍	Expl
∑	Syst
⚖	Team
↪	Share

New features testing

- ⇪ “Big” & “small” features
- 🔍 Domain knowledge > oracle
  - *Several obvious bugs as a side effect*
- ∑ Comprehensive exploratory > project with test cases
  - *The vast majority of bugs*

⇪	Prior
🔍	Expl
∑	Syst
⚖	Team
↪	Share

New features testing

⇪ Prior
🔍 Expl
∑ Syst
⚖ Team
↪ Share

- ⇪ “Big” & “small” features
- 🔍 Domain knowledge > oracle
  - *Several obvious bugs as a side effect*
- ∑ Comprehensive exploratory > project with test cases
  - *The vast majority of bugs*
- ⚖ Talk to developers, reject the acceptance

New features testing

- ⇪ Prior
- 🔍 Expl
- ∑ Syst
- ⚖ Team
- ↪ Share

⇪ “Big” & “small” features

🔍 Domain knowledge > oracle

- ○ *Several obvious bugs as a side effect*

∑ Comprehensive exploratory > project with test cases

- ○ *The vast majority of bugs*

⚖ Talk to developers, reject the acceptance

↪ Regressions, reproductions, acceptance cases

What you do again?

- Release testing
- Reproduce & reduce
- New features testing
- **Fixes verification**
- Infrastructure tasks
- …

Fixes verification

⇪ Prior

🔍 Expl

∑ Syst

⚖️ Team

↪ Share

Fixes verification

- ⇪ Tired of features? Verify!

⇪ Prior
🔍 Expl
∑ Syst
⚖ Team
↪ Share

Fixes verification

⇪ Tired of features? Verify!

🔍 Getting into the context; read the code

⇪ Prior
🔍 Expl
∑ Syst
⚖ Team
↪ Share

Fixes verification

- ⇪ Tired of features? Verify!
- 🔍 Getting into the context; read the code
- ∑ Special search in YouTrack

⇪	Prior
🔍	Expl
∑	Syst
⚖	Team
↪	Share

Fixes verification

- ⇧ Tired of features? Verify!
- 🔍 Getting into the context; read the code
- ∑ Special search in YouTrack
- ⚖ Ask the developers & read the code

⇧	Prior
🔍	Expl
∑	Syst
⚖	Team
↪	Share

Fixes verification

- ⇧ Tired of features? Verify!
- 🔍 Getting into the context; read the code
- ∑ Special search in YouTrack
- ⚖ Ask the developers & read the code
- ↪ Check the related issues and duplicates

⇧	Prior
🔍	Expl
∑	Syst
⚖	Team
↪	Share

What you do again?

- Release testing
- Reproduce & reduce
- New features testing
- Fixes verification
- **Infrastructure tasks**
- …

Infrastructure tasks

⇪ Prior

🔍 Expl

∑ Syst

⚖️ Team

↪ Share

Infrastructure tasks

⇪ When you have some time…

⇪ Prior
🔍 Expl
∑ Syst
⚖️ Team
↪ Share

Infrastructure tasks

⇪ When you have some time…

🔍 Depends on the task

⇪ Prior

🔍 Expl

Σ Syst

⚖ Team

↪ Share

Infrastructure tasks

- ⇪ When you have some time…
- 🔍 Depends on the task
- ∑ Update, automate, tag

⇪ Prior
🔍 Expl
∑ Syst
⚖ Team
↪ Share

Infrastructure tasks

- ⇧ When you have some time…
- 🔍 Depends on the task
- ∑ Update, automate, tag
- ⚖ Race a readiness

⇧ Prior
🔍 Expl
∑ Syst
⚖ Team
↪ Share

Infrastructure tasks

- ⇧ When you have some time…
- 🔍 Depends on the task
- ∑ Update, automate, tag
- ⚖ Race a readiness
- ↪ Sync, share the projects

⇧	Prior
🔍	Expl
∑	Syst
⚖	Team
↪	Share

What you do again?

- Release testing
- Reproduce & reduce
- New features testing
- Fixes verification
- Infrastructure tasks
- ...

...

⇧ Prior

🔍 Expl

∑ Syst

⚖️ Team

↪ Share

...

- ⇪ Background task

⇪	Prior
🔍	Expl
Σ	Syst
⚖️	Team
↪	Share

...

- ⇧ Background task
- 🔍 IDE is mutable
  - *...so are the other dependencies*

⇧	Prior
🔍	Expl
∑	Syst
⚖️	Team
↪	Share

...

- ⇧ Background task
- 🔍 IDE is mutable
  - *...so are the other dependencies*
- ∑ Track the incoming flow & watch github commits

⇧ Prior
🔍 Expl
∑ Syst
⚖️ Team
↪ Share

...

- ⇪ Background task
- 🔍 IDE is mutable
  - *...so are the other dependencies*
- ∑ Track the incoming flow & watch github commits
- ⚖ Read & listen to everything, watch the meetings

⇪ Prior
🔍 Expl
∑ Syst
⚖ Team
↪ Share

...

- ⇪ Background task
- 🔍 IDE is mutable
  - ○ *...so are the other dependencies*
- ∑ Track the incoming flow & watch github commits
- ⚖ Read & listen to everything, watch the meetings
- ↪ Discuss & point out

⇪ Prior
🔍 Expl
∑ Syst
⚖ Team
↪ Share

...

- ⇧ Background task
- 🔍 IDE is mutable
  - ○ *...so are the other dependencies*
- ∑ Track the incoming flow & watch github commits
- ⚖ Read & listen to everything, watch the meetings
- ↪ Discuss & point out
- ⟲ ... :)

⇧	Prior
🔍	Expl
∑	Syst
⚖	Team
↪	Share

What you do again?

- Release testing
- Reproduce & reduce
- New features testing
- Fixes verification
- Infrastructure tasks
- ...

Sum up

- Complex environment

Sum up

- Complex environment
- Intense information flow

Sum up

- Complex environment
- Intense information flow
- Prioritize

Sum up

- Complex environment
- Intense information flow
- Prioritize
- Share the knowledge & work as a team

Sum up

- Complex environment
- Intense information flow
- Prioritize
- Share the knowledge & work as a team
- Be responsive and observable

Kotlin tooling QA

- What are you doing?
- How do you do that?
- **How do they getting into it?**
- Fun?

Newbie onboarding: quick review

Newbie onboarding: quick review

- **Work hard :)**

SLOTHILDA.COM

Newbie onboarding

- Basic infrastructure & process info

Newbie onboarding

- Basic infrastructure & process info
- Kotlin tutorials and documentation (kotl.in)

Newbie onboarding

- Basic infrastructure & process info
- Kotlin tutorials and documentation (kotl.in)
- Trial acceptance

Newbie onboarding

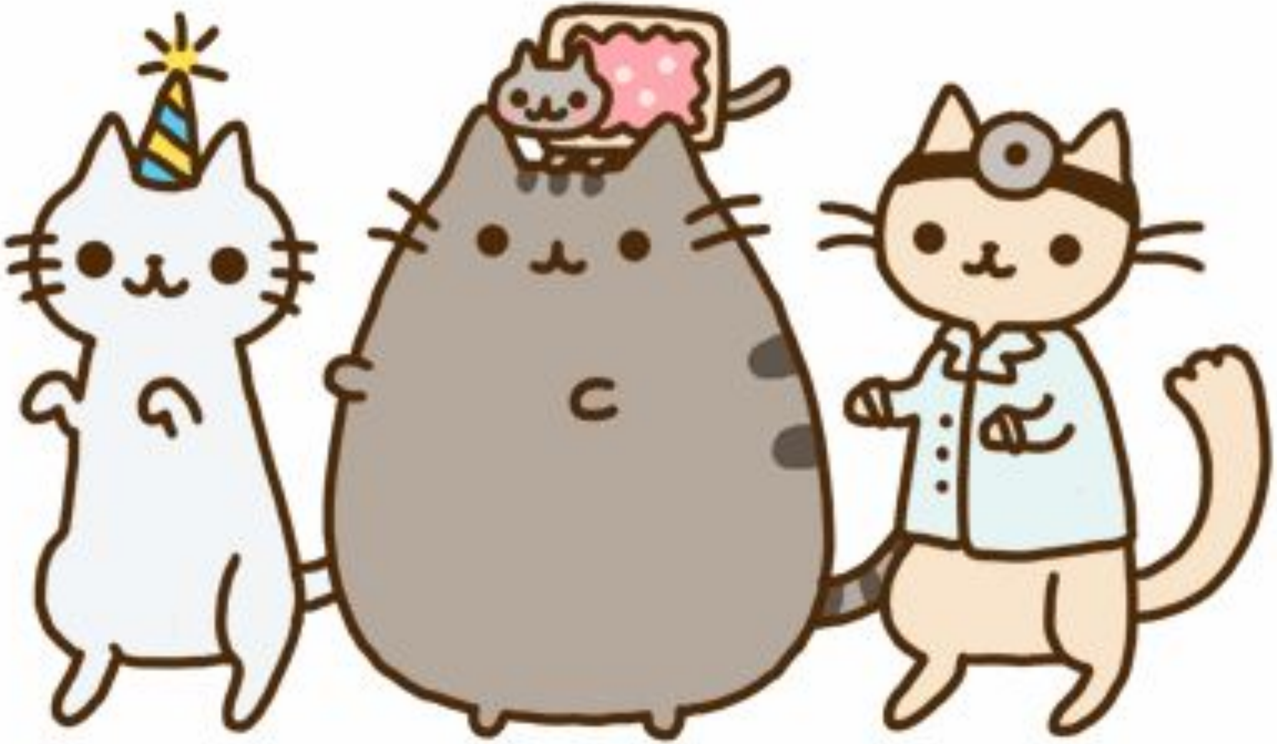
- Basic infrastructure & process info
- Kotlin tutorials and documentation (kotl.in)
- Trial acceptance
- Reproduce, verify

Newbie onboarding

- Basic infrastructure & process info
- Kotlin tutorials and documentation (kotl.in)
- Trial acceptance
- Reproduce, verify
- Test an IDE feature, a language feature

Newbie onboarding

- Basic infrastructure & process info
- Kotlin tutorials and documentation (kotl.in)
- Trial acceptance
- Reproduce, verify
- Test an IDE feature, a language feature
- Get a “big feature”



Why it’s fun?

- The product is cool & valuable

Why it’s fun?

- The product is cool & valuable
- You learn every day

Why it’s fun?

- The product is cool & valuable
- You learn every day
- Very loyal and responsive community

Why it’s fun?

- The product is cool & valuable
- You learn every day
- Very loyal and responsive community
- Bugs are everywhere!

Why it’s fun?

- The product is cool & valuable
- You learn every day
- Very loyal and responsive community
- Bugs are everywhere!
- That was just a part of all activities :)

Thank you!

Liliia Abdulina,

Kotlin tooling QA

kotl.in

Telegram: @wild_lynx

Blog: https://medium.com/@liliia

E-mail: lilia.abdulina@jetbrains.com