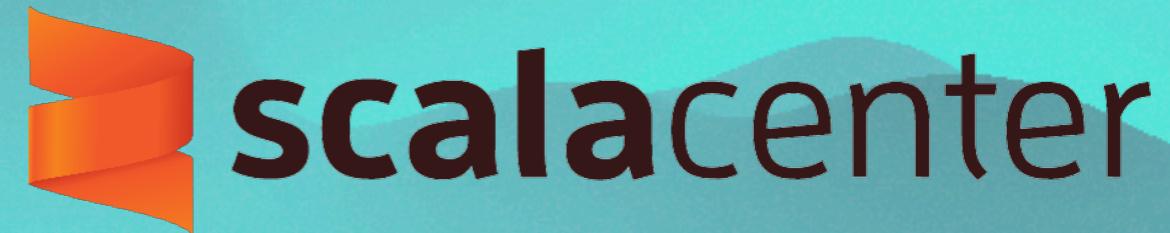


Scastie the Playground of Scala



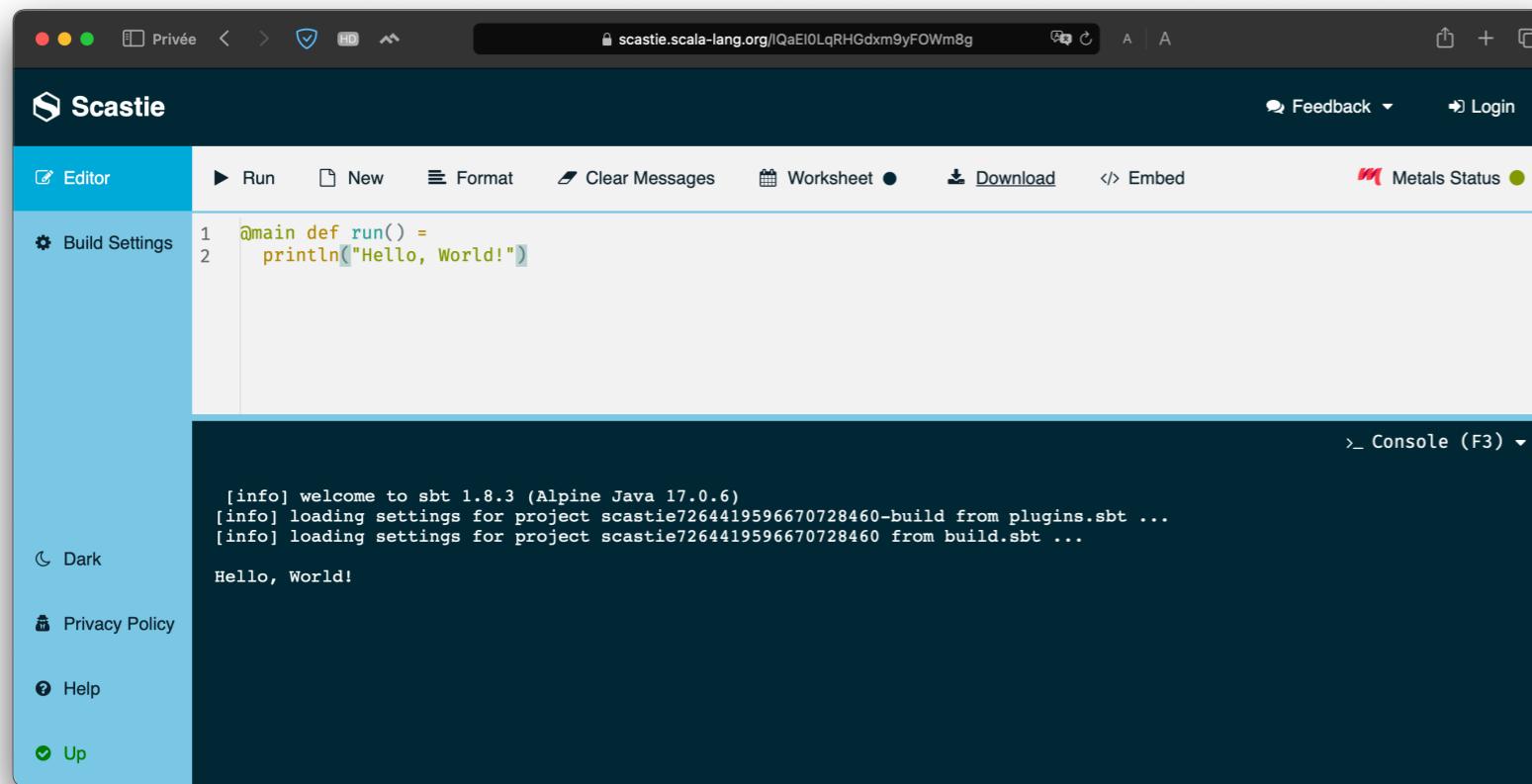
Multiple files support

Timeline

- Project definition
- Challenges
- Front end
- Backend
- Evaluation
- Conclusion & future work

What is Scastie?

- <https://scastie.scala-lang.org/U6hM9iXwT0KiAMH7DJJXmQ>
- <https://github.com/scalacenter/scastie>



Current main features

- Embed feature

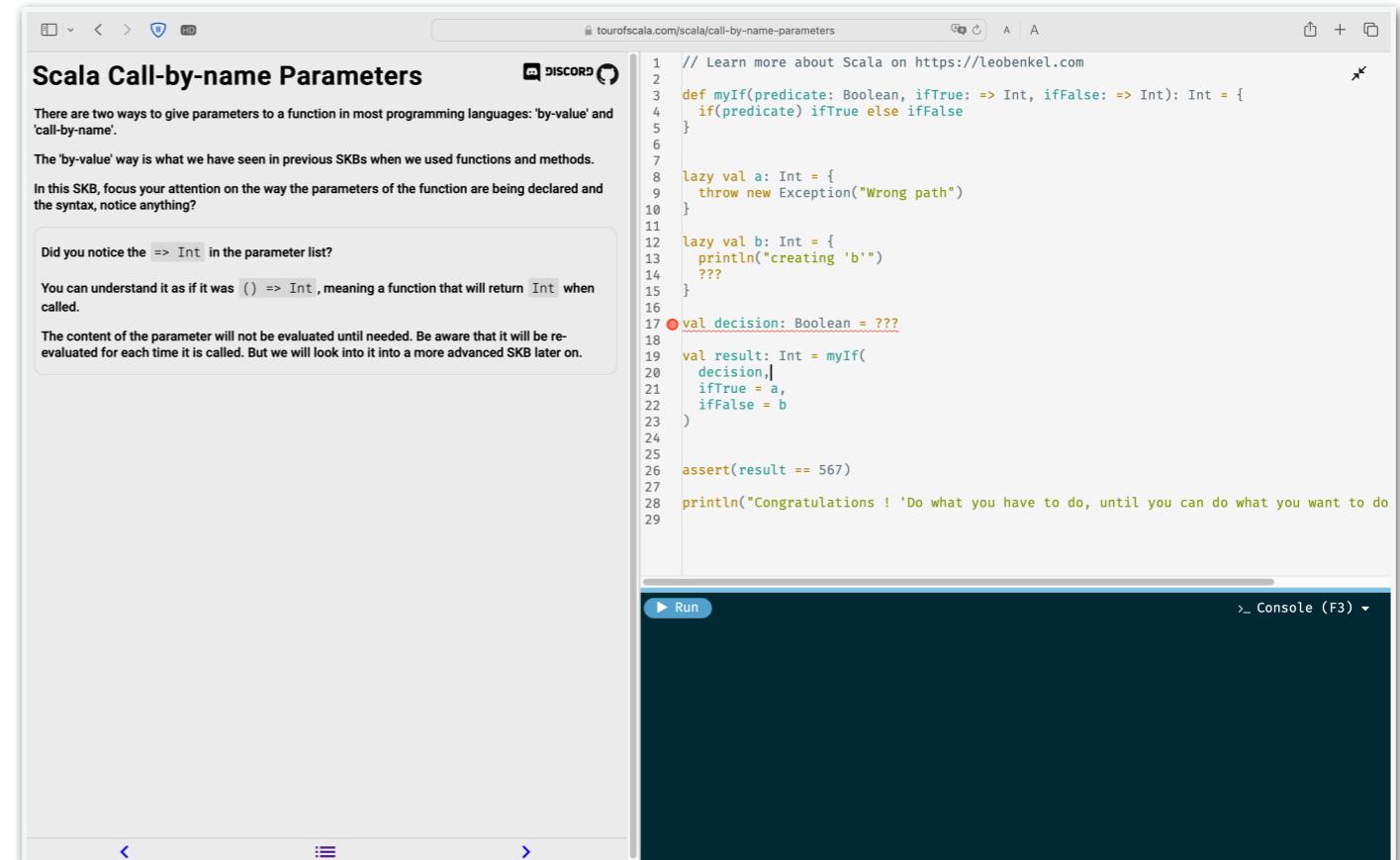
At the request of the OP, here's an [implementation](#) on scastie.

Share Edit Follow Flag

edited Sep 1, 2020 at 19:45

answered Sep 1, 2020 at 6:04

 kfkhaliili
996 ● 1 ● 10 ● 24



The screenshot shows a web browser window with the URL <https://tourofscala.com/scala/call-by-name-parameters>. The page title is "Scala Call-by-name Parameters". It contains a brief explanation of call-by-name parameters and a code editor with the following Scala code:

```
// Learn more about Scala on https://leobenkel.com
def myIf(predicate: Boolean, ifTrue: => Int, ifFalse: => Int): Int = {
  if(predicate) ifTrue else ifFalse
}

lazy val a: Int = {
  throw new Exception("Wrong path")
}

lazy val b: Int = {
  println("creating 'b'")
  ???
}

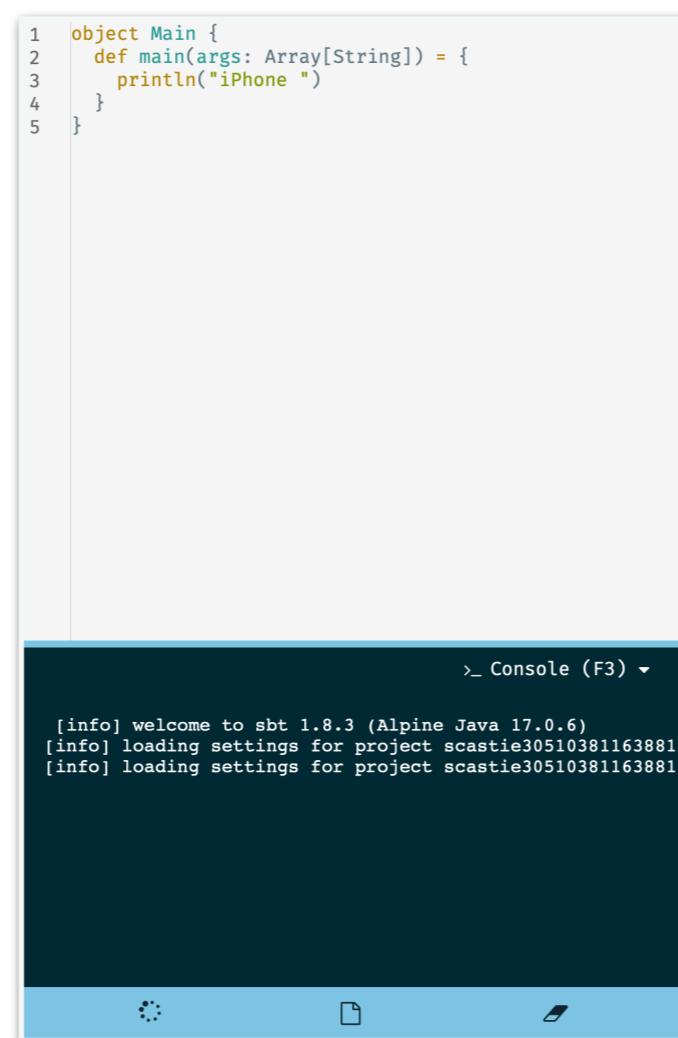
val decision: Boolean = ???
val result: Int = myIf(
  decision,
  ifTrue = a,
  ifFalse = b
)

assert(result == 567)
println("Congratulations ! 'Do what you have to do, until you can do what you want to do'"
```

Below the code editor is a terminal window titled "Console (F3)" with a "Run" button.

Current main features

- Mobile/Tablet modes



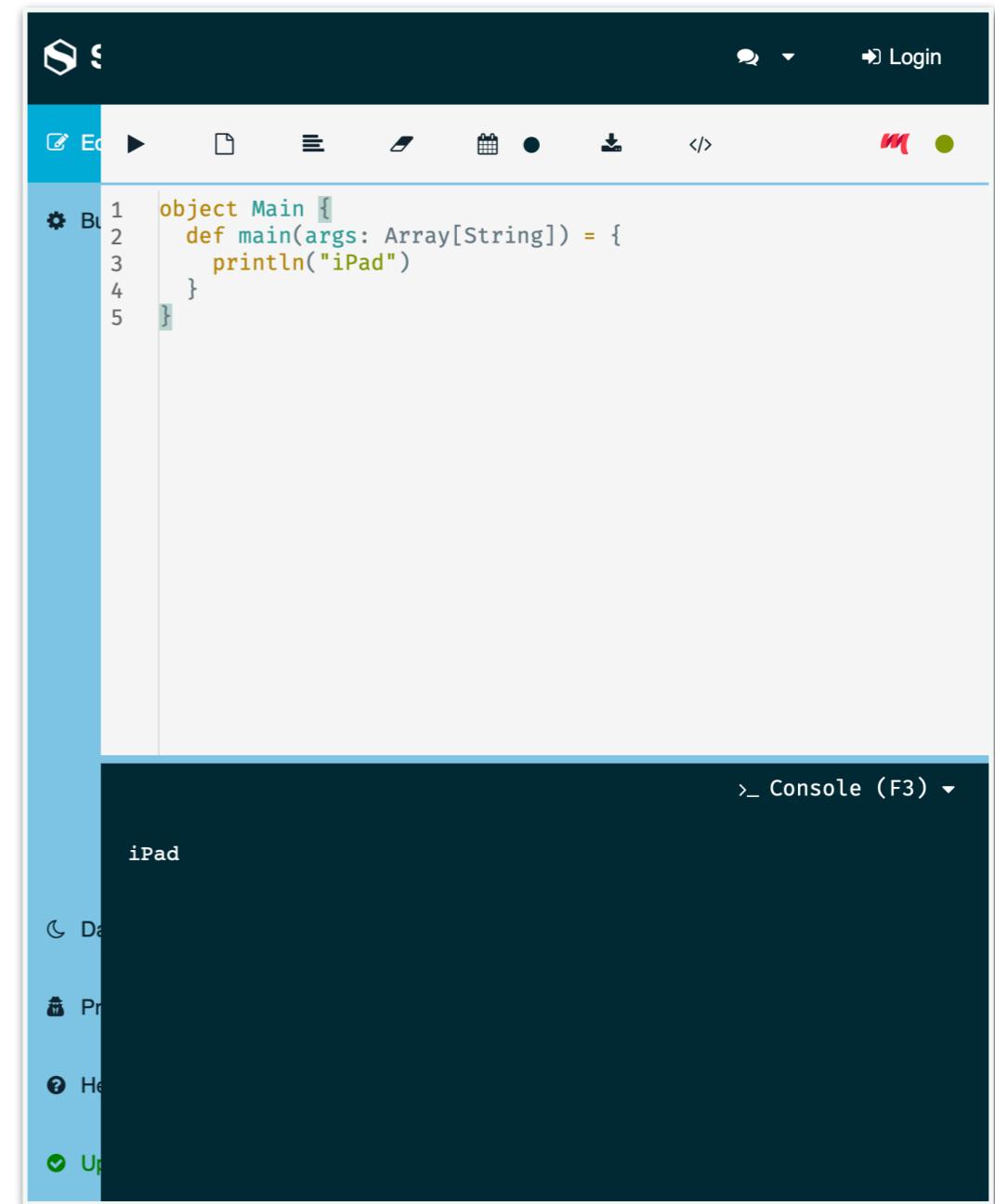
A screenshot of the Scastie mobile application. At the top is a header bar with the Scastie logo and a search icon. Below the header is a toolbar with icons for file operations like Open, Save, and Print. The main area contains a code editor with the following Scala code:

```
object Main {  
  def main(args: Array[String]) = {  
    println("iPhone ")  
  }  
}
```

Below the code editor is a dark-themed console window with the title "Console (F3)". It displays the output of the Scala code:

```
[info] welcome to sbt 1.8.3 (Alpine Java 17.0.6)  
[info] loading settings for project scastie305103811638818  
[info] loading settings for project scastie305103811638818
```

The bottom of the screen features a blue navigation bar with icons for back, forward, and search.



A screenshot of the Scastie mobile application. At the top is a header bar with the Scastie logo and a search icon. Below the header is a toolbar with icons for file operations like Open, Save, and Print. The main area contains a code editor with the same Scala code as the first screenshot:

```
object Main {  
  def main(args: Array[String]) = {  
    println("iPad")  
  }  
}
```

Below the code editor is a dark-themed console window with the title "Console (F3)". It displays the output of the Scala code:

```
iPad
```

The bottom of the screen features a blue navigation bar with icons for back, forward, and search.

Current main features

- Worksheet mode

Scastie

Editor Run New Format Clear Messages Worksheet Download Embed

Build Settings

```
1 val x = 5
2
3 "Hello World".substring(x)  World: java.lang.String
4
5 x+1  6: scala.Int
6
7 val l = List(1, 2, 6, 7, 8, 10)
8 l.filter(_ % 2 == 0)  List(2, 6, 8, 10): scala.collection.immutable.List[scala.Int]
```

Instrumentation

Scastie

Editor Run New Format Clear Messages Worksheet Download

Build Settings

```
1 val width = 600
2 html"<img src=\"https://ichef.bbci.co.uk/news/1024/cpsprodpb/5616/production/1024x576.jpg\" alt=\"OceanGate TITAN submersible\" width=" + width + " height=" + width + "/>"
```

Titan submersible



Porthole window Rear cover Landing frame

Source: OceanGate Expeditions

Dark Privacy Policy Help Up

HTML tags

Problem

The screenshot shows the Scastie web-based Scala code editor interface. The main area displays a Scala program with two classes: `Kangaroo` and `Human`. The `Kangaroo` class has methods for jumping and eating, which print messages to the console. The `Human` class has a method for walking, which also prints a message. The code is annotated with line numbers from 1 to 28.

```
3  class Kangaroo(val name: String) {
4    var energy: Int = 100
5
6    def jump(): Unit = {
7      if (energy >= 10) {
8        println(s"$name is jumping!")
9        energy -= 10
10    } else {
11      println(s"$name is too tired to jump.")
12    }
13  }
14
15  def eat(): Unit = {
16    println(s"$name is eating.")
17    energy += 20
18  }
19
20
21  class Human(val name: String) {
22    var hunger: Int = 50
23
24    def walk(): Unit = {
25      if (hunger <= 70) {
26        println(s"$name is walking.")
27        hunger += 10
28      } else {
```

The console output shows the simulation starting and then completing with the message "Simulation complete!".

```
Starting the Kangaroo Simulation!
-----
Kangy is jumping!
John is walking.
Kangy is eating.
John is eating.
-----
Simulation complete!
```

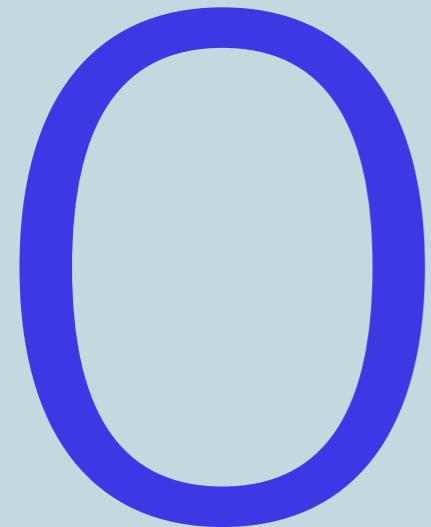
The sidebar on the left includes links for "Dark" mode, "Privacy Policy", "Help", and "Up". The top navigation bar includes "Feedback", "DavidKalajdzic", "Metals Status", and other standard browser controls.

Project goal

- **Multi file support**
 - organise and navigate big snippets
 - Macros support
- Develop UI to edit multiple files (file hierarchy/tabs)
 - File/Folder creation/deletion/renaming/moving
 - Handling tab switching (Error messages per file)
- Server side logic for compilation
- Enable autocomplete across files
- Backward compatibility of all existing features
- Scastie is becoming an IDE...

48

.scala files in one folder



Documentation

Front end

- ScalaJS-React (<https://github.com/japgolly/scalajs-react>)
- Shared API definitions with Backend
- Challenges
 - Refactor the UI
 - FileHierarchy, TabStrip connected
 - Don't destroy existing features

Design a layout

A screenshot of the Scastie Scala code editor. The interface includes a top navigation bar with a logo, a search bar, and a 'Login' button. Below the navigation is a toolbar with icons for file operations like 'Edit', 'Run', and 'Download'. The main workspace shows a Scala code editor with the following code:

```
object Main {
  def main(args: Array[String]) = {
    println("iPad")
  }
}
```

At the bottom of the editor is a 'Console' window displaying the output: "iPad". A sidebar on the left contains links for 'Documentation', 'Privacy Policy', 'Help', and 'Upcoming'.

A screenshot of the Scastie Scala code editor. The interface features a sidebar on the left with a file tree showing files like 'Main.scala', 'Kangaroo.scala', 'Cat.scala', 'Dog.scala', 'House.scala', 'Building.scala', and 'Human.scala'. The main workspace has two tabs open: 'Main.scala' and 'Kangaroo.scala'. The 'Kangaroo.scala' tab contains the following code:

```
object Kangaroo {
  def jump(height: Int) = {
    println(s"Ｋ is jumping $height meters!!")
  }
}
```

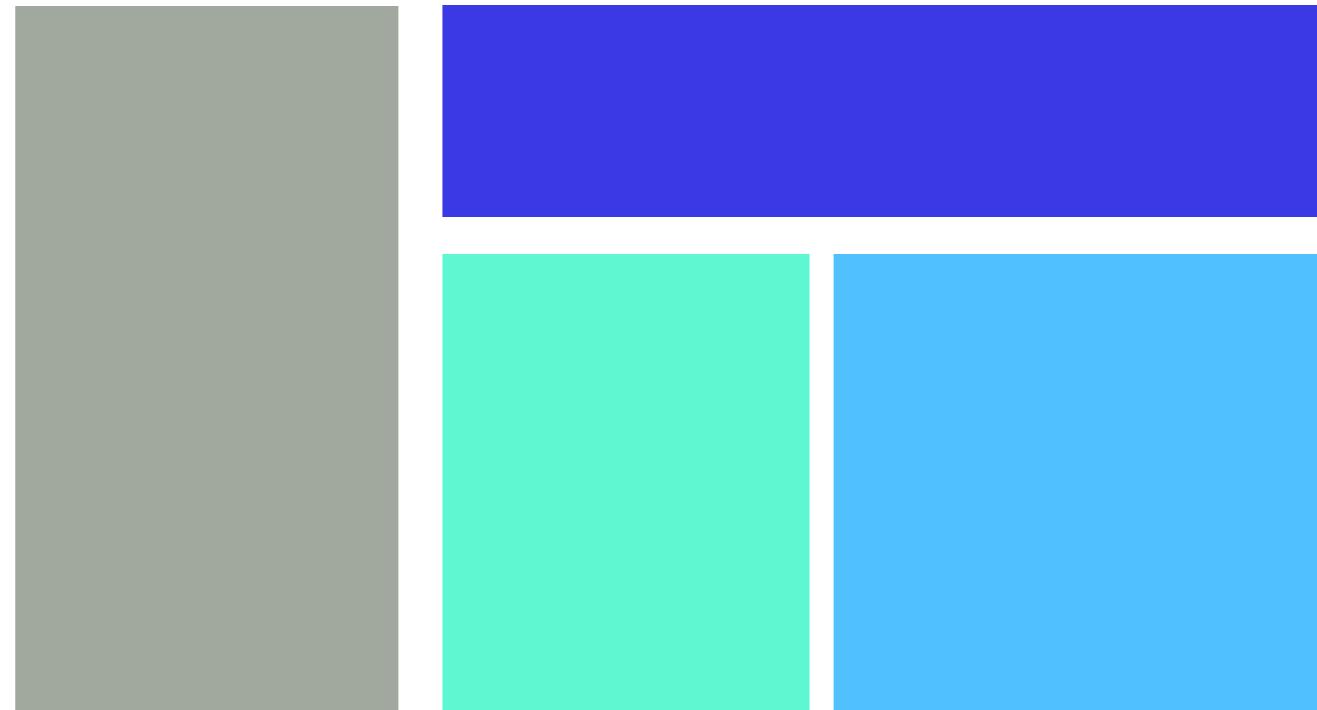
The 'Console' window at the bottom shows the output: "Ｋ is jumping 10 meters!!". The top navigation bar includes a 'Feedback' button, a user profile for 'DavidKalajdzic', and a 'Metals Status' indicator. The bottom of the screen shows a footer with links for 'About', 'Privacy Policy', and 'Help'.

CSS Grid vs Flexbox

- Flexible box



- Grid layout



React technology

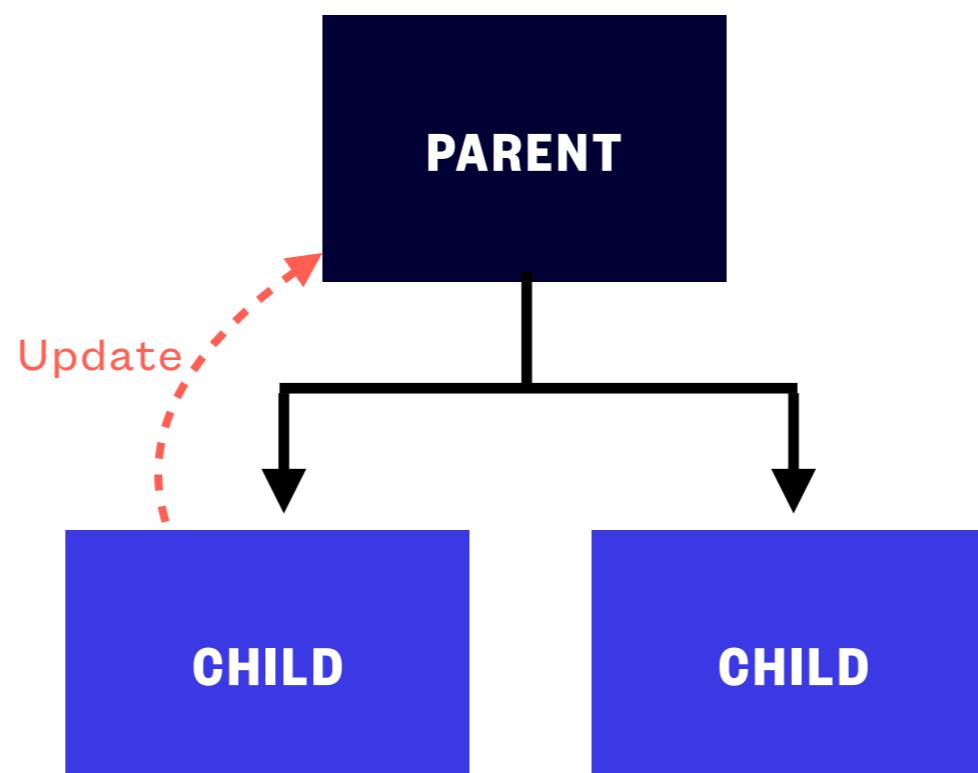
- Allows to write less code
- Reusable components (single responsibility principle)
- Components encapsulates State+UI behaviour
- Refreshing performant



Search...	
<input type="checkbox"/>	Only show products in stock
Name Price	
Sporting Goods	
Football	\$49.99
Baseball	\$9.99
Basketball	\$29.99
Electronics	
iPod Touch	\$99.99
iPhone 5	\$399.99
Nexus 7	\$199.99

Example of complex UI (1)

React technology



React tree (Parent as source of truth)

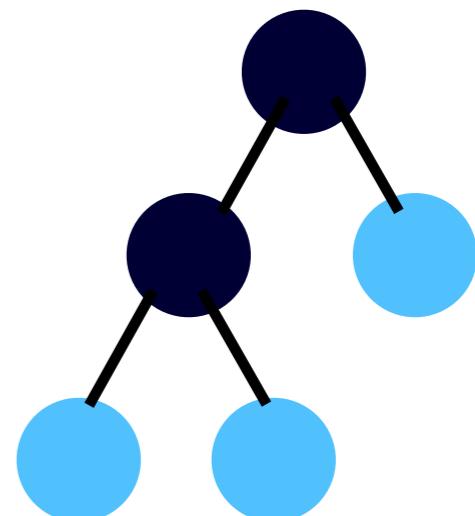
Functional programming

- In React we don't directly mutate vars
- We call "modState"
- Functional language enforces this concept

```
sealed trait FileOrFolder(name, path)
```

```
case class Folder(name, path, children: List[FileOrFolder])
```

```
case class File(name, path, content: String)
```



Demo

Front end conclusions

- Refactor UI to incorporate FileHierarchy & TabStrip
- Mobile & Tablet support
- Thin side bar (fast switching)
- Footer bar
- Embed mode (file hierarchy & tabstrip)
- Presentation mode (tab strip hidden)

Compose as needed

The image displays three side-by-side screenshots of a Scala development environment, likely IntelliJ IDEA, illustrating modular code composition.

Left Screenshot: Shows the `Main.scala` file with the following code:

```
object Main {
  def main(args: Array[String]) = {
    val h = 10
    Kangaroo.jump(h)
  }
}
```

The project structure on the left includes files like `Kangaroo.scala`, `Cat.scala`, `Dog.scala`, `House.scala`, `Building.scala`, and `Human.scala`.

Middle Screenshot: Shows the `House.scala` file with the following code:

```
object Main {
  def main(args: Array[String]) = {
    val h = 10
    Kangaroo.jump(h)
  }
}
```

The project structure on the left includes files like `Main.scala`, `animals` (containing `Kangaroo.scala`), `city` (containing `House.scala`), and `Building.scala`.

Right Screenshot: Shows the `Kangaroo.scala` file with the following code:

```
object Kangaroo {
  def jump(height: Int): Unit = {
    println(s" Kangaroo is jumping $height meters!!")
  }

  def kick(power: Int): Unit = {
    println(s" Kangaroo is kicking with power level $power!!")
  }

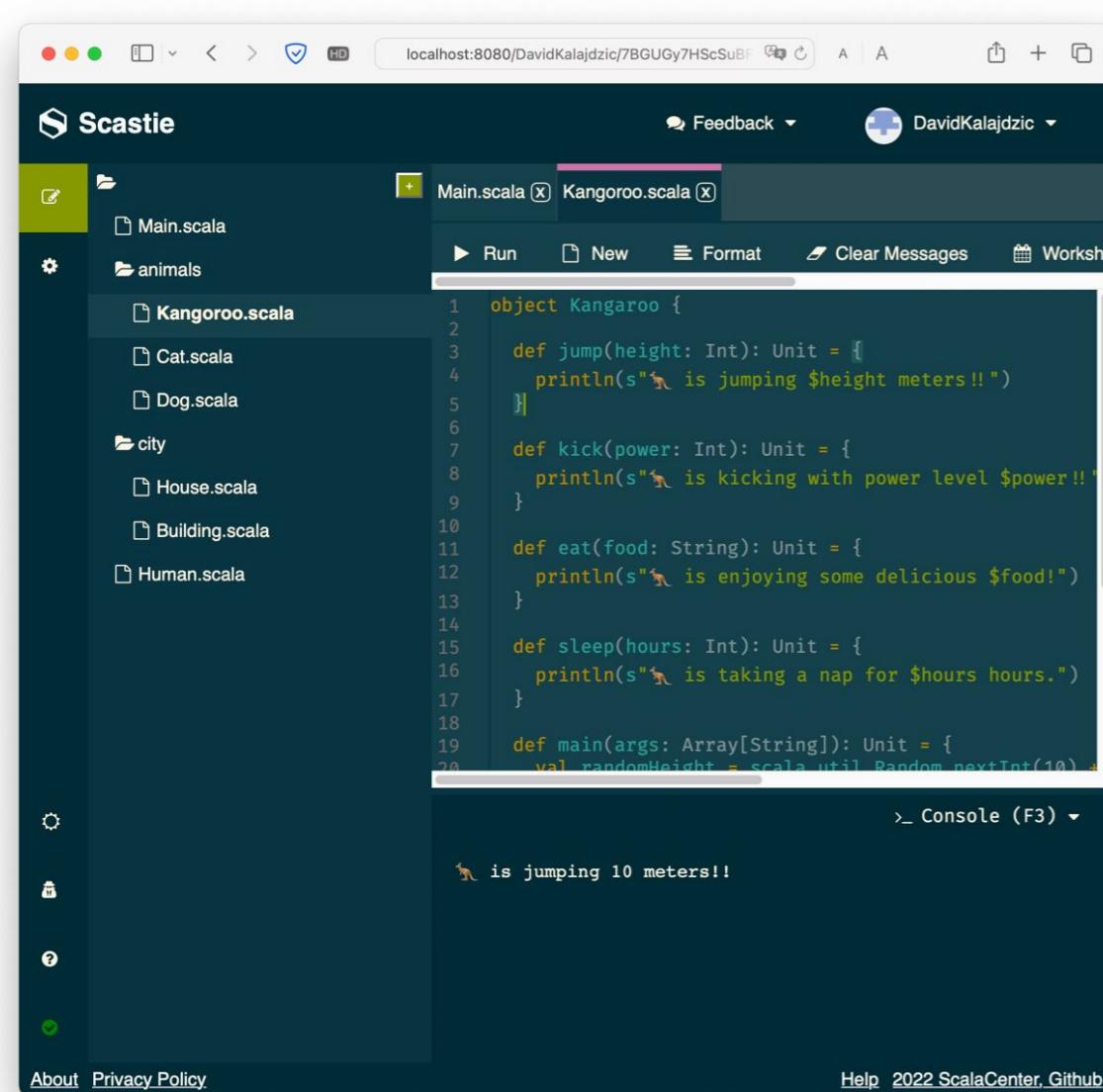
  def eat(food: String): Unit = {
    println(s" Kangaroo is enjoying some delicious $food!")
  }

  def sleep(hours: Int): Unit = {
    println(s" Kangaroo is taking a nap for $hours hours.")
  }
}
```

The project structure on the left includes files like `Main.scala`, `animals` (containing `Kangaroo.scala`), `city` (containing `House.scala`), `Building.scala`, and `Human.scala`.

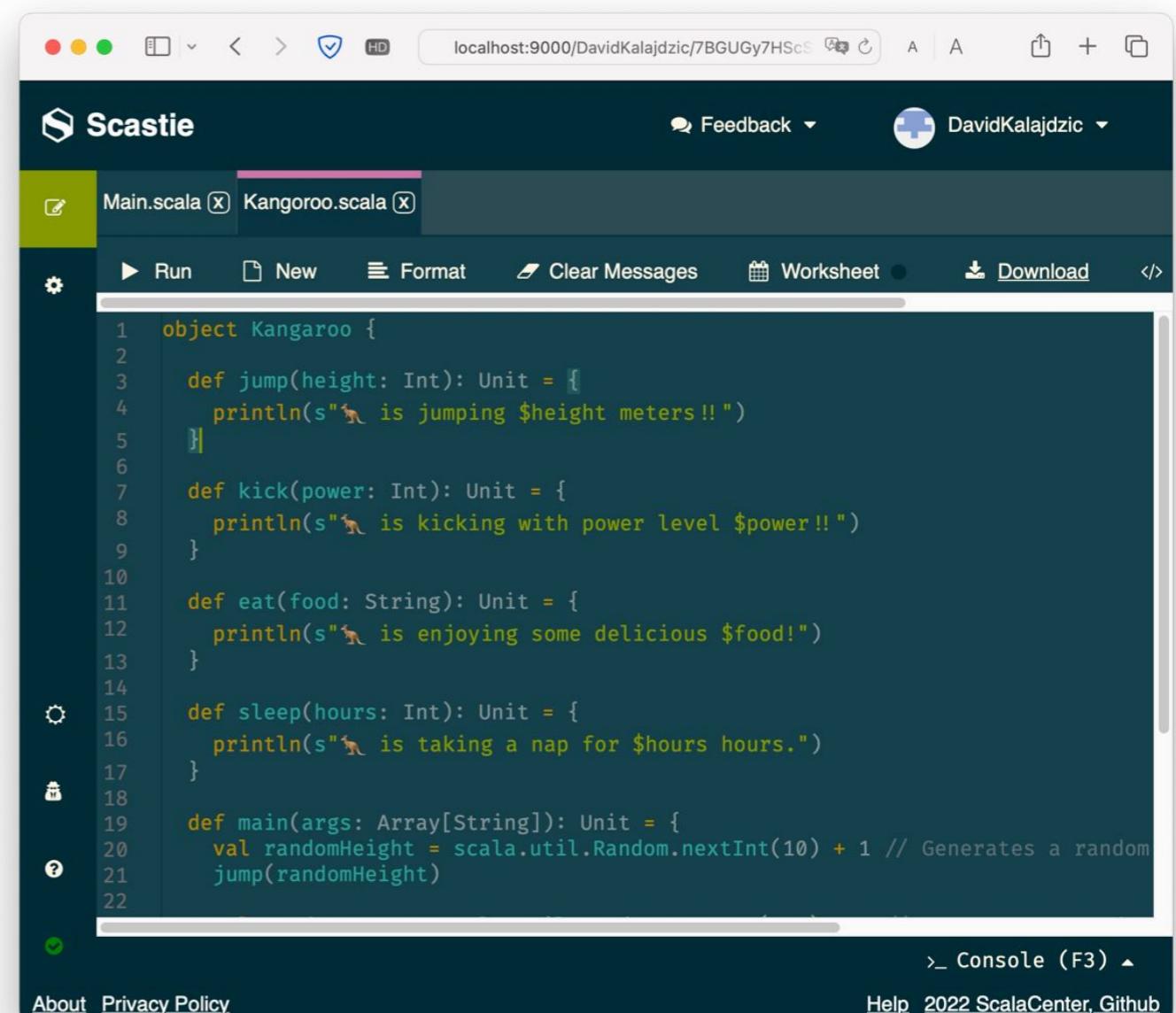
Each screenshot shows a terminal window at the bottom displaying the output of the `jump` method: `Kangaroo is jumping 10 meters!!`

Scroll bars width



A screenshot of the Scastie web application interface. The main window title is "Scastie". The URL in the address bar is "localhost:8080/DavidKalajdzic/7BGUGy7HScSuBF". The code editor shows two files: "Main.scala" and "Kangoroo.scala". The "Kangoroo.scala" file contains Scala code for a Kangaroo object with methods like jump, kick, eat, and sleep. The "Console" output shows the result of running the code. The scroll bar on the right side of the code editor is relatively narrow.

```
object Kangaroo {  
    def jump(height: Int): Unit = {  
        println(s" Kangaroo is jumping $height meters!!")  
    }  
  
    def kick(power: Int): Unit = {  
        println(s" Kangaroo is kicking with power level $power!!")  
    }  
  
    def eat(food: String): Unit = {  
        println(s" Kangaroo is enjoying some delicious $food!")  
    }  
  
    def sleep(hours: Int): Unit = {  
        println(s" Kangaroo is taking a nap for $hours hours.")  
    }  
  
    def main(args: Array[String]): Unit = {  
        val randomHeight = scala.util.Random.nextInt(10) + 1 // Generates a random  
    }  
}  
  
Kangaroo is jumping 10 meters!!
```



A screenshot of the Scastie web application interface, similar to the one above but with a wider scroll bar on the right side of the code editor. The URL in the address bar is "localhost:9000/DavidKalajdzic/7BGUGy7HScSuBF". The code editor shows the same "Main.scala" and "Kangoroo.scala" files. The "Kangoroo.scala" file contains the same Scala code. The "Console" output shows the result of running the code. The scroll bar is significantly wider than in the first screenshot.

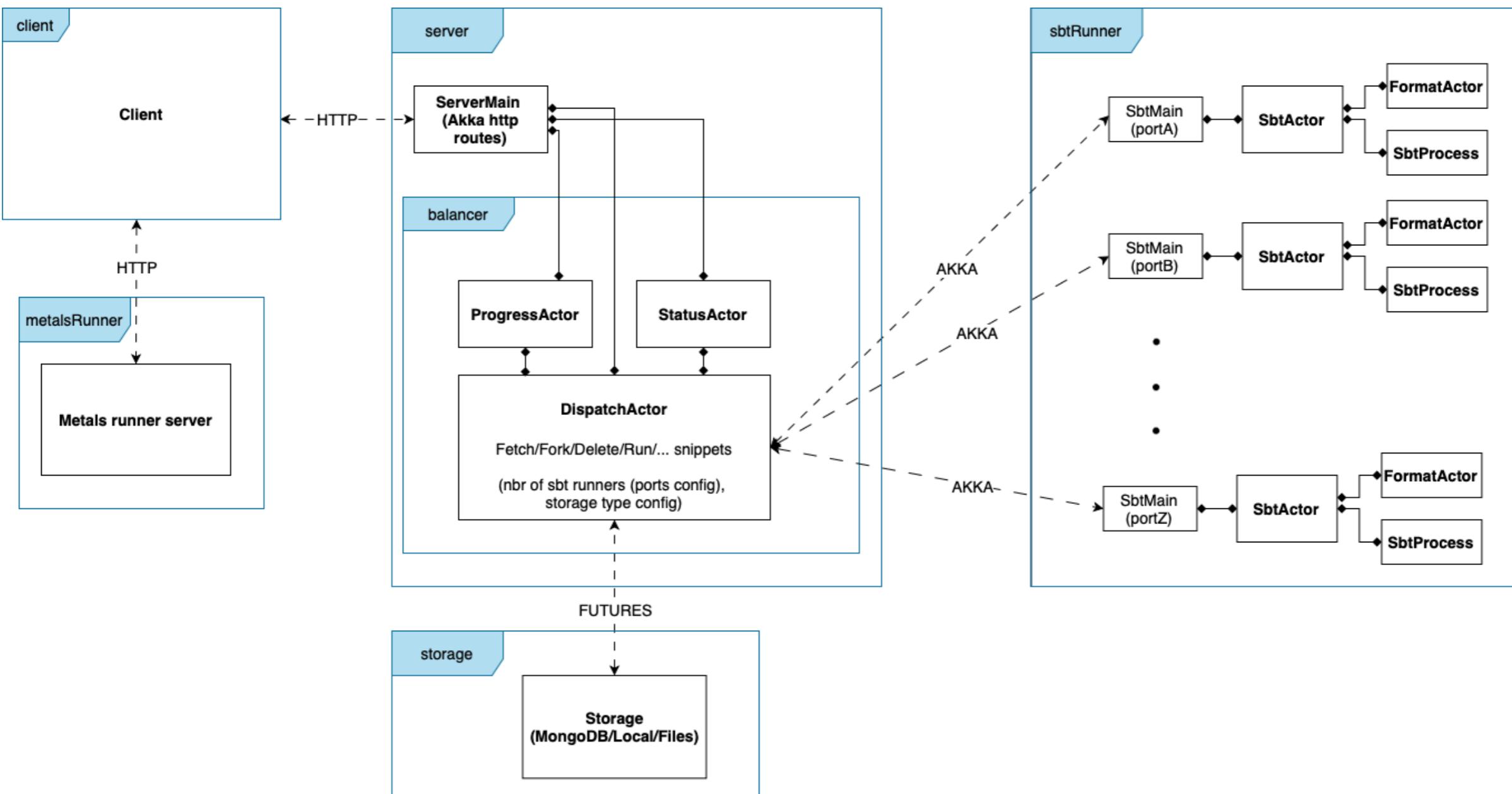
```
object Kangaroo {  
    def jump(height: Int): Unit = {  
        println(s" Kangaroo is jumping $height meters!!")  
    }  
  
    def kick(power: Int): Unit = {  
        println(s" Kangaroo is kicking with power level $power!!")  
    }  
  
    def eat(food: String): Unit = {  
        println(s" Kangaroo is enjoying some delicious $food!")  
    }  
  
    def sleep(hours: Int): Unit = {  
        println(s" Kangaroo is taking a nap for $hours hours.")  
    }  
  
    def main(args: Array[String]): Unit = {  
        val randomHeight = scala.util.Random.nextInt(10) + 1 // Generates a random  
    }  
}
```

Backend

Main challenge

- Support old snippets and features
- Define a data structure for Multiple files

Backend architecture



Support old snippets

- Change schema & write Migration script
 - + clean and simple solution
 - + major feature hardcoded
 - – migration scripts to write
- Handling versions on the fly
 - + easy to implement for highly demanded services
 - – annoying to maintain
 - – increased latency

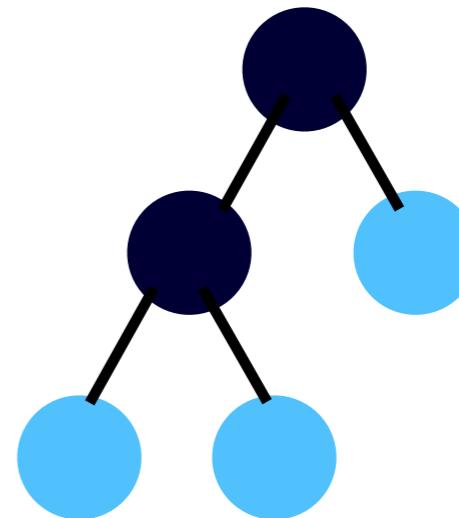
Migration script

- String → Folder

```
sealed trait FileOrFolder(name, path)
```

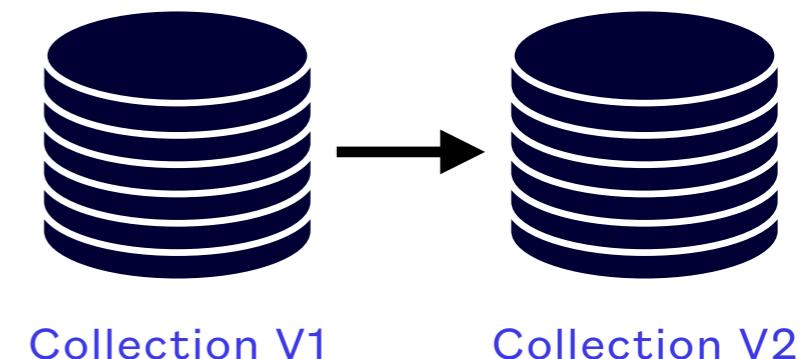
```
case class Folder(name, path, children: List[FileOrFolder])
```

```
case class File(name, path, content: String)
```



- Migration script

- create new collection (with same indexes)
- fetch all snippet ids
- transform each snippet and store in that new collection



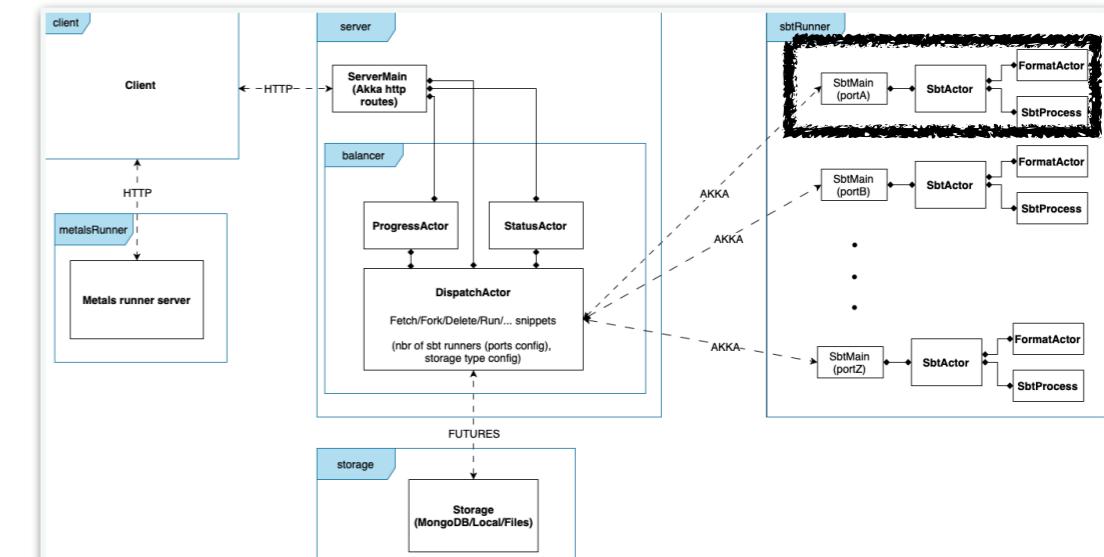
SbtProcess

- SbtProcess actor allows interaction in command line with a sbt instance
- A sbt project is created on disk from a Scastie snippet
- It was overwriting a main.scala
- *reload; compile/compileInputs*

- *fgRun*

```
Music          javafx-sdk-11.0.2      untitled1
Parallels      jpp
Pictures       nltk_data
david:~$ pwd
/Users/david
david:~$ cd aaaaaa/scastie
david:~/aaaaaa/scastie (multiple-files) $ sbt
[info] welcome to sbt 1.8.2 (Amazon.com Inc., Java 17.0.6)
[info] loading global plugins from /Users/david/.sbt/1.0/plugins
[info] loading settings for project scastie-build-build from metals.sbt ...
[info] loading project definition from /Users/david/aaaaaa/scastie/project/project/project
[info] loading settings for project scastie-build-build from metals.sbt ...
[info] loading project definition from /Users/david/aaaaaa/scastie/project/project/project
[success] Generated .bloop/scastie-build-build.json
[success] Total time: 0 s, completed 29 juin 2023, 02:41:51
[info] loading settings for project scastie-build-build from metals.sbt,plugins.sbt ...
[info] loading project definition from /Users/david/aaaaaa/scastie/project/project
[success] Generated .bloop/api-sbt.json
[success] Generated .bloop/scastie-build.json
[success] Total time: 1 s, completed 29 juin 2023, 02:41:52
[info] loading settings for project scastie-build-build from build.sbt,metals.sbt,plugins.sbt ...
[info] loading project definition from /Users/david/aaaaaa/scastie/project
[success] Generated .bloop/api-sbt.json
[success] Generated .bloop/scastie-build.json
[success] Total time: 4 s, completed 29 juin 2023, 02:41:57
[info] loading settings for project scastie from build.sbt ...
[info] resolving key references (24233 settings) ...
[info] set current project to scastie (in build file:/Users/david/aaaaaa/scastie/)
[info] sbt server started at local://Users/david/.sbt/1.0/server/be5ff2eb2ffe5117fcf9.sock
[info] started sbt server
sbt:scastie:
```

sbt illustration



Recall the architecture

Compilation & Runtime infos

```
1 object Main {  
2     def main(args: Array[String]) = {  
3         println("Hello World")  
4     unclosed string literal  
5     }  
6 }
```

Compile time error

```
1 object Main {  
2     def main(args: Array[String]) = {  
3         println(10/0)  
4     java.lang.ArithmetricException: / by zero  
5     at Main$.main(main.scala:3)  
6     at Main.main(main.scala)  
at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)  
at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:77)  
at java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)  
at java.base/java.lang.reflect.Method.invoke(Method.java:568)  
at sbt.Run.invokeMain(Run.scala:143)  
at sbt.Run.execute$1(Run.scala:93)  
at sbt.Run.$anonfun$runWithLoader$5(Run.scala:120)  
at sbt.Run$.executeSuccess(Run.scala:186)  
at sbt.Run.runWithLoader(Run.scala:120)  
at sbt.Run.run(Run.scala:127)  
at com.olegchy.scastie.sbtscastie.SbtScastiePlugin$$anon$1.$anonfun$run$1(SbtScastiePlugin.scala:38)  
at scala.runtime.java8.JFunction0$mcV$sp.apply(JFunction0$mcV$sp.java:23)  
at sbt.util.InterfaceUtil$$anon$1.get(InterfaceUtil.scala:21)  
at sbt.ScastieTrapExit$App.run(ScastieTrapExit.scala:258)  
at java.base/java.lang.Thread.run(Thread.java:833)
```

Runtime error

Sbt-scastie plugin

```
private def sbtPluginsConfig0(withSbtScastie: Boolean): String = {
    val targetConfig = target.sbtPluginsConfig

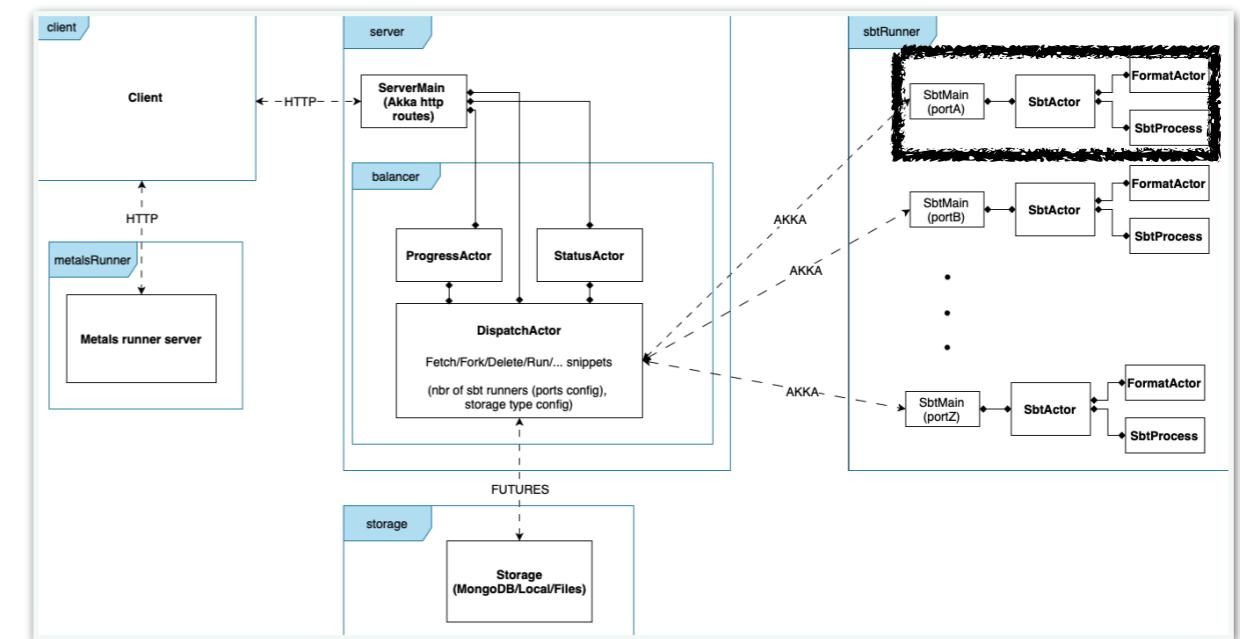
    val sbtScastie =
        if (withSbtScastie)
            s"""addSbtPlugin("org.scastie" % "sbt-scastie" % "${BuildInfo.versionRuntime}")"""
        else ""

    mapToConfig(targetConfig, sbtScastie)
}
```

sbt-scastie Plugin injected to each sbtRunner

```
sbtRunner < 182: Input(fgRun)
sbtRunner -- Running --
sbtRunner > 183 30767ms: {"output":{"line":"[info] compiling 1 Scala source to /private/var/folders/jb/f9h5g4rs3xb_sdsm_dbbr9jw0000gn/T/scastie7455467925025399069/src/main/scala/Main.scala.o"}}
sbtRunner > 184 102ms: [{"severity":"Error","line":3,"filePath":"$BASE/src/main/scala/code.scala","message":"unclosed string literal"}]
sbtRunner > 185 10ms: {"output":{"line":"[error] (Compile / compileIncremental) Compilation failed","tpe":"StdOut"},"tpe":"SbtOutput"}
sbtRunner > 186 9ms:
sbtRunner > 187 11ms: OtiIyNIWuc
sbtRunner -- Ready --
```

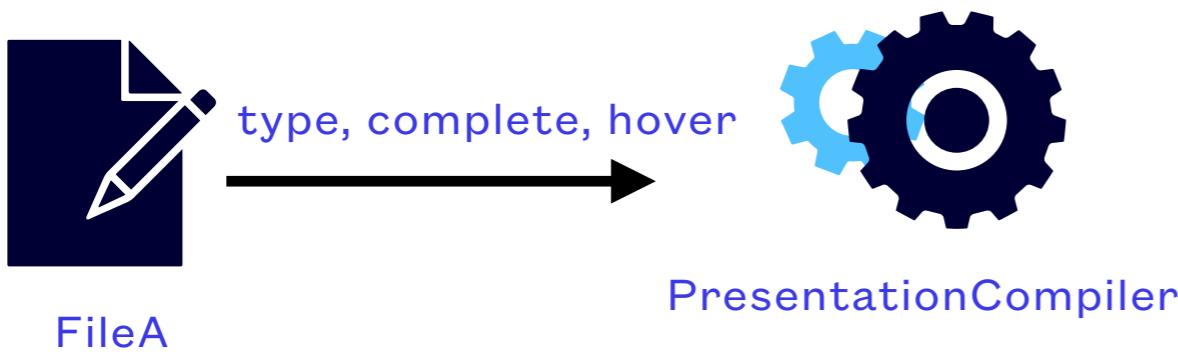
```
sbtRunner Running: (login: Some(DavidKalajdzic), ip: unknown)
sbtRunner Folder(List(File(Main.scala.o))
sbtRunner writing in /var/folders/jb/f9h5g4rs3xb_sdsm_dbbr9jw0000gn/T/scastie7455467925025399069/src/main/scala
sbtRunner -- Running --
sbtRunner < 158: Input(fgRun)
sbtRunner > 159 21809ms: {"output":{"line":"[info] compiling 1 Scala source to /private/var/folders/jb/f9h5g4rs3xb_sdsm_dbbr9jw0000gn/T/scastie7455467925025399069/src/main/scala/Main.scala.o"}}
sbtRunner > 160 239ms: {"output":{"line":"[info] done compiling","tpe":"StdOut"},"tpe":"SbtOutput"}
sbtRunner > 161 16ms: {"output":{"line":"[info] running Main","tpe":"StdOut"},"tpe":"SbtOutput"}
sbtRunner > 162 4ms: Hello World
sbtRunner > 163 1ms: {"output":{"line":"[error] (run-main-0) java.lang.ArithmaticException: / by zero","tpe":"StdOut"},"tpe":"SbtOutput"} <1 internal lines>
sbtRunner > 165 1ms: {"output":{"line":"[error] java.lang.ArithmaticException: / by zero","tpe":"StdOut"},"tpe":"SbtOutput"}
sbtRunner > 166 0ms: {"output":{"line":"[error] \tat Main$.main(Main.scala:4)","tpe":"StdOut"},"tpe":"SbtOutput"}
sbtRunner > 167 0ms: {"output":{"line":"[error] \tat Main.main(Main.scala)","tpe":"StdOut"},"tpe":"SbtOutput"} <4 internal lines>
sbtRunner > 172 28ms: {"output":{"line":"[error] Nonzero exit code: 1","tpe":"StdOut"},"tpe":"SbtOutput"}
sbtRunner > 173 0ms: {"output":{"line":"[error] (Compile / fgRun) Nonzero exit code: 1","tpe":"StdOut"},"tpe":"SbtOutput"}
sbtRunner > 174 10ms:
sbtRunner > 175 9ms: OtiIyNIWuc
sbtRunner -- Ready --
server c.o.scastie.balancer.DispatchActor:98 | pinged ActorSelection[Anchor(akka://SbtRunner@127.0.0.1:5150/), Path(/user/SbtActor)] server
```



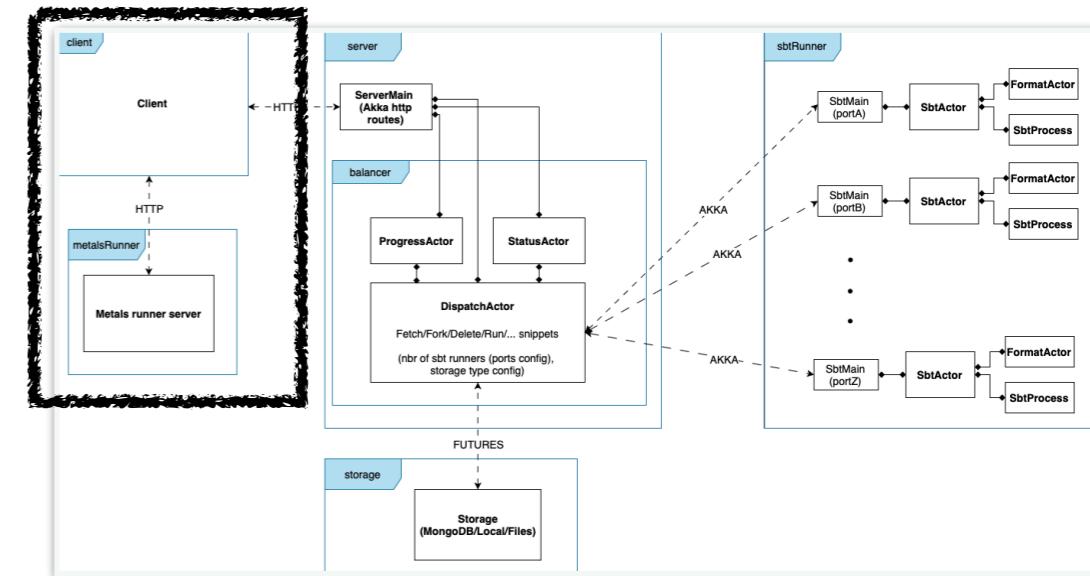
sbtRunner

Autocompletion (ScastieMetals)

- Standalone running server (no knowledge on files, just ongoing edits)
- PresentationCompiler (a faster async version of the Scala Compiler)
- After each keystroke, reanalyse

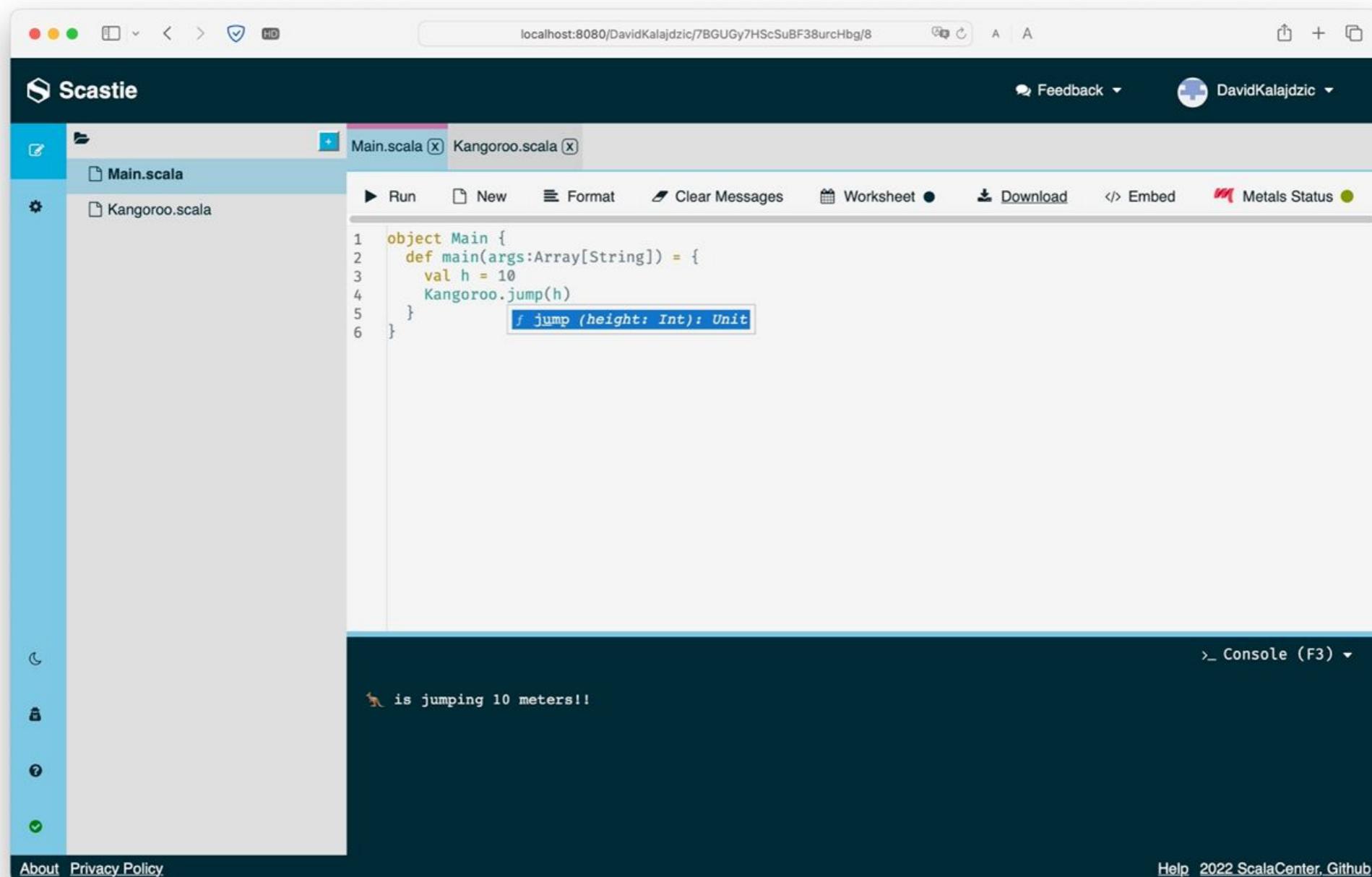


- Previously :
 - Virtual file with an URI of NoSourceFile.path
 - /complete, /hover



Recall the architecture

Autocompletion in action



The screenshot shows the Scastie Scala code editor interface. The top bar displays the URL `localhost:8080/DavidKalajdzic/7BGUGy7HScSuBF38urcHbg/8`. The main workspace has two tabs: `Main.scala` and `Kangoroo.scala`. The `Main.scala` tab contains the following code:object Main {
 def main(args:Array[String]) = {
 val h = 10
 Kangoroo.jump(h)
 }
}

A tooltip or dropdown menu is open over the word `jump`, showing the method signature `f jump (height: Int): Unit`. Below the code editor is a dark blue console window displaying the output of the program:

```
is jumping 10 meters!!
```

Keeping the extras

- Export snippet as Zip
- Code formatting the current file

Conclusion

Conclusion

- Evaluation & testing to protect those new features
- Compatibility with browsers
- Scastie is a real world application (attention to details for prod)
- Backward compatibility of snippets
- Multi file snippets will enhance user experience
- Macros are now supported

