Direct Embedding

Joris Beau

EPFL - LAMP
Project in computer science
Directed by **Martin Odersky**, Supervised by **Vojin Jovanovic**

January 15, 2015

Outline

Motivation

- Instigated by Slick database library
- Embedding DSLs simply!
- Provide a painless logic for reification
- Direct embedding has avantages:
  - better error messages
  - compile-time

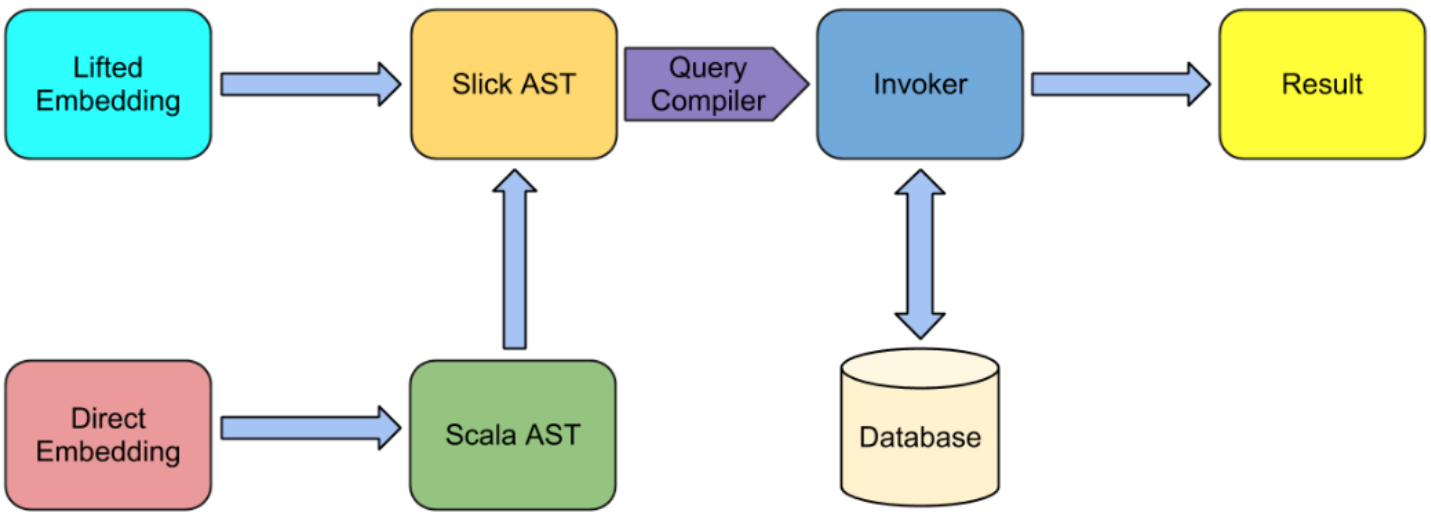
Goal

1. Educational aspect
2. Implement a core prototype
3. Make it work
4. Extend it
5. Might evolve to a library

Features

- Take care of macro
- Handle Intermediate Representation (IR)
- $\Longrightarrow$ users code in Scala-like language in Scala projects
- $\Longrightarrow$ users' friendly

Big Picture of Slick



Direct vs lifted

Direct

- AST generated compile-time
- macro based
- Scala type
- at runtime, errors for unsupported methods
- experimental on Slick

Lifted

- changes into a deep DSL IR
- operator overloading
- deep type
- caught at compilation unsupported operators
- used by Slick, LMS

Reification: Need to modify Scala AST

Beginning

How to modify Scala AST?

Beginning

How to modify Scala AST?
→ **quasiquote?**

Beginning

No quasiquote

but ...

Beginning

but

we can access to the symbols → @ANNOTATIONS

Beginning

@ANNOTATIONS

```
@reifyAs(ClassCons)
class ClassExample {
  ???
}
```

```
@table(name="COFFEES")
  case class Coffee(
    ...
  )
```

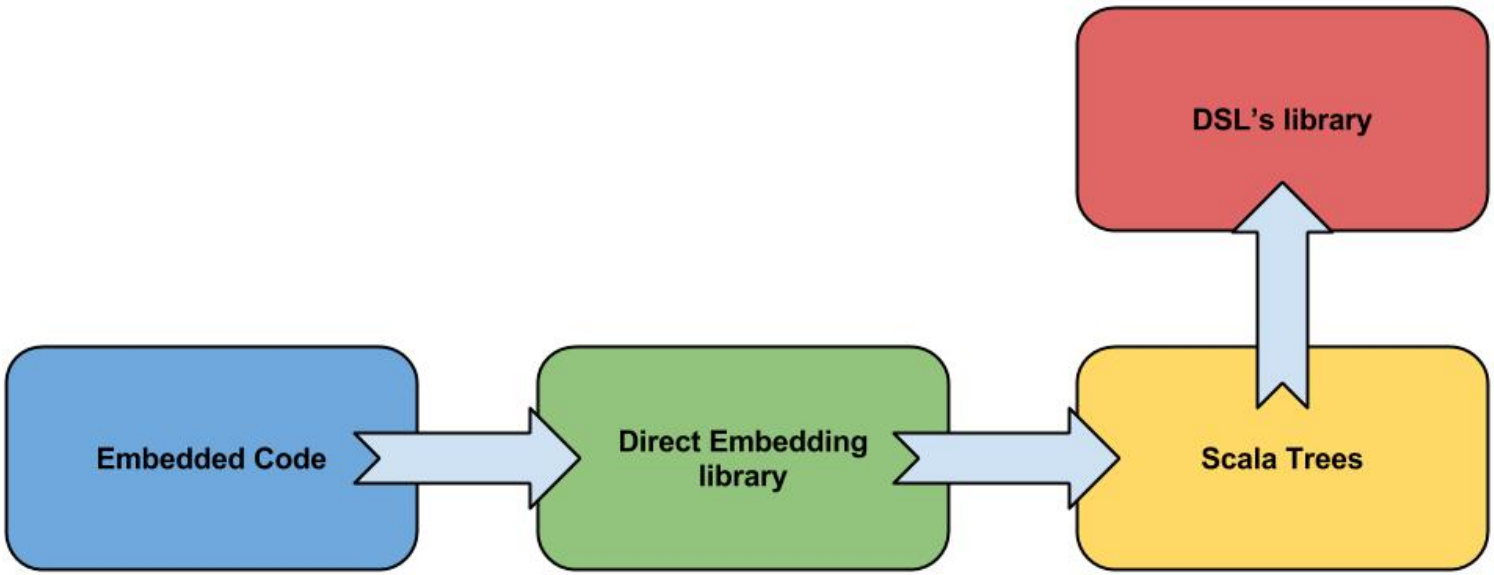
Objects

Cases	Object	Nested	Classes
val value	✓	✓	✓
def foo	✓	✓	✓
def foo(args)	✓	✓	✓
def foo[T, U]: (T, U)	✓	✓	✓
def foo[T, U](t: T, u: U): (T, U)	✓	✓	✓
def foo[T](t_1: T)(...)(t_a: T)	✓	✓	✓

Language specification

if	X
while	X
do while	X
lazy val	X
return	X

Overview


DSL's library
Embedded Code
Direct Embedding library
Scala Trees

An example: embedded code

Annotate

```
@reifyAs(JustArgs)
def justArgs(x: Int): Int = ???
```

Lift

```
lift {
  ObjectExample.justArgs(1)
}
```

An example: tree

Tree

ObjectExample.justArgs(1)
  annotated
directembedding.reifyAs(JustArgs)

Raw tree

Apply(Select(Ident(ch.epfl.directembedding.test.ObjectExample), TermName("justArgs")), List(Literal(Constant(1))))

An example: macro

From symbol, arguments, type, we can use the annotation to reify the tree:

Macro

$\Longrightarrow$ macro q"..."

Returns

JustArgs.apply(1)

An example: Summary

- Users accordingly annotate DSLs expressions
- ASTs generated and transformed via macro
- After the modification, the tree contains the DSL representation .i.e. the annotation

Comparison

Direct

```
@table(name="COFFEES")
  case class Coffee(
    ...
  )
```

Lifted

```
object Coffees extends Table[...]("COFFEES") {
    ...
}
```

Going further

- Operator
- Recursion
- Raw block of code

Demonstration

Thank you

Special thanks