

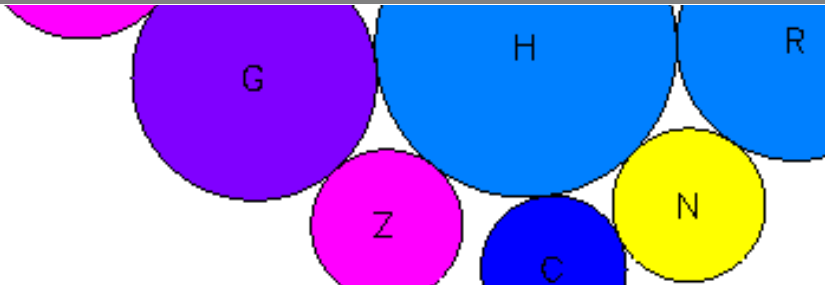
A Haskell Roadshow

Joachim Breitner

The Karlsruhe Functional Programmers Meetup Group

December 18, 2012

PROGRAMMING PARADIGMS GROUP



Haskell is known for

being a

functional
pure
lazy evaluated
strongly typed
interpreted
compiled

programming language ...

Haskell is known for

being a

functional
pure
lazy evaluated
strongly typed
interpreted
compiled

programming language ...

and it is **fun** to program in.

Live demonstration

Did our promise hold?

Haskell is indeed

functional ?

pure

lazy evaluated

strongly typed

interpreted

compiled

programming language ...

and it is **fun** to program in.

Did our promise hold?

Haskell is indeed

functional ✓
pure ?
lazy evaluated
strongly typed
interpreted
compiled

programming language ...

and it is **fun** to program in.

Did our promise hold?

Haskell is indeed

functional ✓
pure ✓
lazy evaluated ?
strongly typed
interpreted
compiled

programming language ...

and it is **fun** to program in.

Did our promise hold?

Haskell is indeed

functional ✓
pure ✓
lazy evaluated ✓
strongly typed ?
interpreted
compiled

programming language ...

and it is **fun** to program in.

Did our promise hold?

Haskell is indeed

functional ✓
pure ✓
lazy evaluated ✓
strongly typed ✓
interpreted ?
compiled

programming language ...

and it is **fun** to program in.

Did our promise hold?

Haskell is indeed

functional ✓
pure ✓
lazy evaluated ✓
strongly typed ✓
interpreted ✓
compiled ?

programming language ...

and it is **fun** to program in.

Did our promise hold?

Haskell is indeed

functional	✓
pure	✓
lazy evaluated	✓
strongly typed	✓
interpreted	✓
compiled	✓

programming language ...

and it is **fun** to program in. ?

What we skipped today

All the small things...

- More about data types
- (Many) more benefits from the type system
- Polymorphism
- Type classes
- Monads
- Foreign Function Interface

What we skipped today

All the small things...

- More about data types
- (Many) more benefits from the type system
- Polymorphism
- Type classes
- Monads
- Foreign Function Interface

... you will find here

- Tutorial “Learn you a Haskell”
- O’Reilly book “Real World Haskell”
- Tutorial “Write Yourself a Scheme in 48 Hours”

Conclusion

Writing Haskell code

- takes less time,
- produces less bugs and
- is more fun.

Therefore, CU all on

[#haskell](#) on IRC (freenode)

and on the

haskell-cafe@haskell.org mailing list!

© 2012 Joachim Breitner.

Distributed under the terms of the Creative Commons Attribution license.