Java is not just an island, it’s dynamic!

Christopher Probst,
Matthias Hesse

Institut für Informatik
Heinrich-Heine-Universität Düsseldorf

22.04.2015

Content

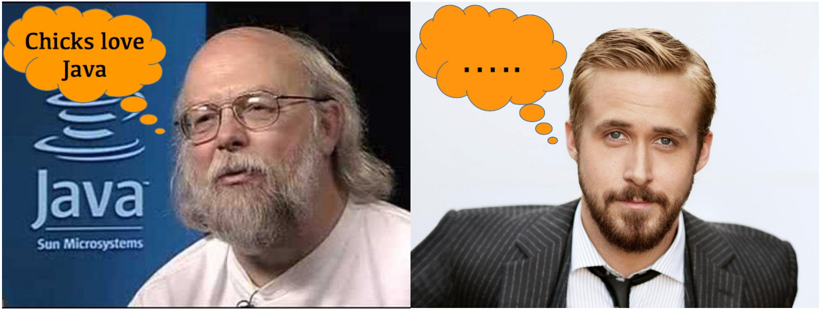
$ java -history

$ java -features

$ java -is-dynamic

$ java -history

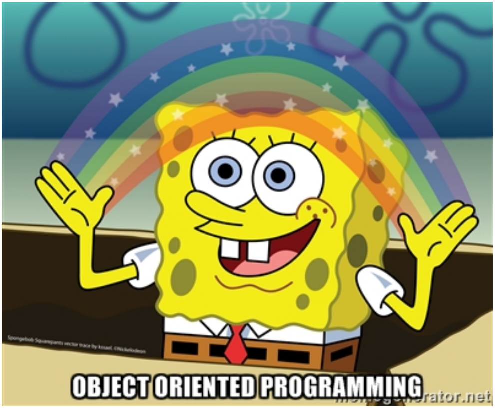
- Created by James Gosling in 1995 (Sun Microsystems, later Oracle)



- Typing: strong (no implicit casting, mostly), static (variables have types)
- Cross-platform (kind of...)
- Paradigms: object-oriented, structured, imperative, functional, generic, reflective, concurrent

$ java -feature object-orientation

- Java is object-oriented, so everything is/should be an object!


Spongebob Squarepants vector trace by kesael. ©Nickelodeon
OBJECT ORIENTED PROGRAMMING
memegenerator.net

$ java -feature object-orientation

- Except byte, boolean, short, char, int, long, float, double
- They are primitive (of course...)

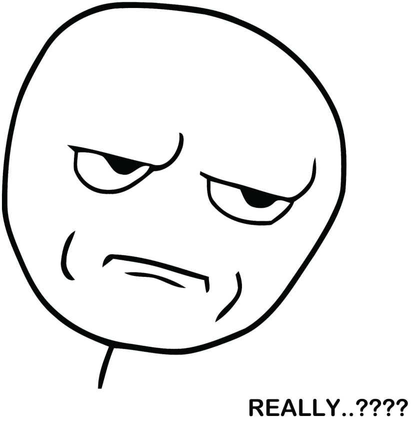
$ java -feature generics

Java has generics:

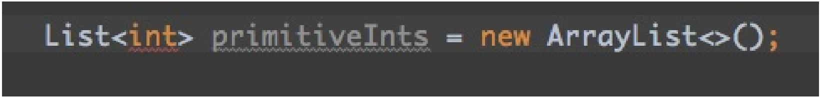
- Useful to write generic algorithms
- Implement container classes (lists, maps, etc.)
- Generic compile errors can be hard to solve, but not as bad as C++

$ java -feature generics

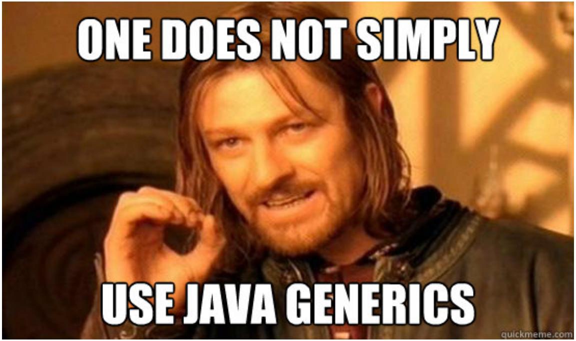
Drawback: Only works with reference types =(



$ java -feature generics



```
List<int> primitiveInts = new ArrayList<>();
```



$ java -feature generics

- Might change with Java 9 (2016) or Java 10 (2018)
- HotSpot JEPs & JSRs: Project Valhalla, Project Panama

$ java -minor-features

- Code can only live in classes
- Syntactically close to C/C++ (Braces, etc.)
- Reflective
- Built-in support for multithreading (synchronized keyword, etc.)
- Built-in support for serialization (transient keyword, etc.)
- No multiple inheritance, all methods are virtual and overloadable
- Since JDK8: Native support for lambda’s and functional programming =)

$ java -minor-features

- Well defined concurrency libraries, arguably the best ones out there
- Very performant network library implementation
- Therefore heavily used in backend systems (financial, social media, etc.)
- Large number of open-source libraries
- Gigantic community

$ java -feature hotspot

- Oracle's reference implementation is the HotSpot JVM
- Byte code interpreter
- Just-in-time compilation (state-of-the-art, gets better in Java 9)
- Garbage-collection (state-of-the-art, even concurrent)
- Optimized for concurrency
- Other VMs: Dalvik, ART, RoboVM, Azul Systems, etc.

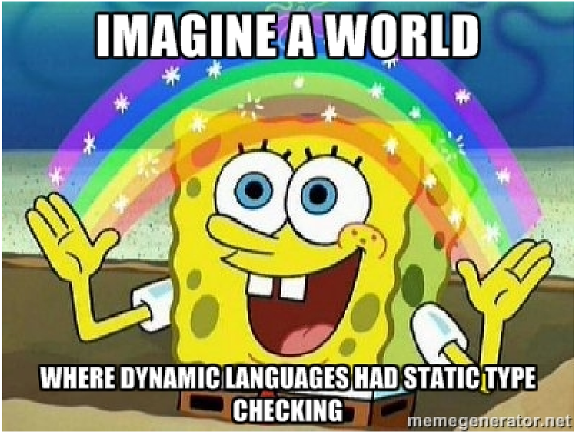
$ java -feature hotspot

Runtime characteristics:

- It takes a while to warm up the JIT
- Learns about code-flow, dead-code elimination
- Dynamic stack allocation (through escape-analysis)
- Dynamic code optimizations
- At some point very, very fast
- Relatively high memory usage
- Arguably on-par with C/C++ (no flame-war intended)
- Conclusion: Perfect for long running, backend server applications

$ java -is-dynamic

But can Java be considered a dynamic language ?



$ java -is-dynamic

"The term dynamic programming language describes a class of programming languages that share a number of common runtime characteristics that are available in static languages only during compilation, if at all.[. . .]"

"[. . .]These behaviors can include the ability to extend the currently running program [. . .] even by modifying the internals of the language itself, all during program execution. While these behaviors can be emulated in almost any language [. . .] such behaviors are integral, built-in characteristics of dynamic languages."

T. Mikkonen and A. Taivalsaari, "Using JavaScript as a real programming language" 2007.

$ java -is-dynamic

So, can Java »extend a currently running program«, maybe »even by modifying the internals of the language itself, all during program execution«?

- Java reflection can modify certain aspects during runtime
- The Java class loader can load source code during runtime
- Javassist (Library) can create/modify Java classes during runtime
- But: It’s a workaround (a hack)
- So it’s not an »integral, built-in characteristic« of Java

$ java -is-dynamic

Dynamic check list

- Interactive (JavaREPL, wait for demo)
- Everything is an object (almost)
- Dynamic Typing (Simulated by using Object, wait for demo)
- Most things changeable at run-time (Yes, but it’s hacky)
- Reflection (Yes!)
- Late-Bound Everything (Simulated by using Object, wait for demo)
- Garbage Collected (Yes!)
- Interpreted (Yes!)

$ java -conclusion

- Java is almost a dynamic language
- It can be stretched alot, but it hurts
- Not meant to be used this way
- Not necessarily a bad thing
- Java sits between Python and C++
- However, a lot of dynamic languages compile down to Java byte code!
- Clojure, Jython, JRuby, and A LOT of others

$ java -audience

Thanks for listening!