KotlinConf

Cords and gumballs

Mike Hearn
mike@r3.com

Who what why huh?!


corda

Who am I?

Kotlin early adopter: first patch to Kotlin website Sept 2014, introduced to my first product Feb 2015.

corda Lead Platform Engineer on Corda

Google Senior engineer at Google (7.5 yrs)

bitcoinj Bitcoin

- Early user (2009, four months after release)
- Long term developer (2010-2015)
- Wrote **bitcoinj**, widely used Java library

r3.

What does Corda do?

Improve how **businesses** work together …

… by replacing **message-oriented** workflows …

… with a shared, world wide, **distributed database** …

… that **nobody owns** and which has no administrators.

Applications to **finance**, **healthcare**, **oil & gas**, **cargo shipping**, for Bitcoin style **consumer e-cash**, and **supply chain integrity**, and maybe more stuff we didn't think of yet.

r3.

Who + what are we?

- Precise nature of what Corda is?
  Not a topic for today!

- Quite interesting computer science though:

  - Bitcoin inspired peer to peer protocol
  - Lots of cryptography. Intel SGX memory encryption and hardware security.
  - Serialised coroutines to implement business processes
  - Fully deterministic version of the JVM
  - Sophisticated identity infrastructure
  - Large type safe API for solving business coordination problems
  - Open source: **corda.net**

Corda and Kotlin

Vital stats

117,708 lines of Kotlin

25+ developers

471 Corda graduates

9,760 commits

2 years old tomorrow!

94.1%
Kotlin

5.2%
Java

Why Kotlin?

- Started using Kotlin **before** 1.0 shipped.
- Very risky move!
- But a calculated risk. It worked.
- No regrets. Would do it again.
- Better devs, happier devs, easier to hire.

r3.

Things that rocked

- Everyone loves Kotlin!
- Mainstream acceptance came fast!
- Some new hires considered us *specifically* because we used Kotlin.
- Bank developers often use it too, although it's optional
- Dokka has a JavaDoc skin!! 😍
- We use DSLs. We use TornadoFX.

Unique challenges

- Creating a **large** Java API under huge time pressure
- Kotlin's Java interop is *very good*
  (… but not perfect)
- Very early adopters
- We use Quasar for Java compatible continuations
- None of the team knew Kotlin before (except me)

r3.

The catch?

- First 18 months were a fight against IDE exceptions.
- Making *perfect* Java API still has caveats.
- IntelliJ is amazing but most team members don't know the best tricks!
- People trip on advanced generics issues ~once per week
- We can't use Kotlin continuations because we must support Java.
  We can't use Kotlin serialisation for the same reason.
  We can't use Kotlin/Native for the same reason.

r3.

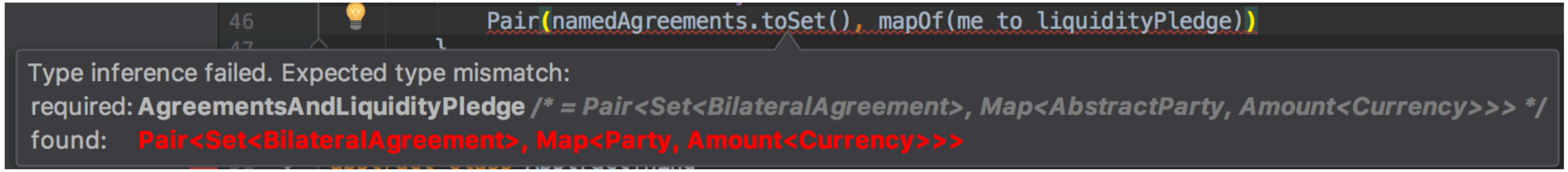
Thus, most big new JetBrains efforts don't help us.

Getting specific

Example mistakes

- Missing `@JvmOverloads` annotations.
- Missing `@JvmStatic` annotations.
- Companion objects polluting the API.
- `@param` instead of `@property` in Kdocs
- Forgetting to make stuff private.
- Can't suppress internal packages in Dokka yet.
- What does internal visibility do, anyway?

Example issues with generics



mpawlak 7:22 PM
added and commented on this JavaScript/JSON snippet

```
1  Error:(34, 36) Kotlin: Type inference failed: Cannot infer type
   parameter Q in constructor OrComposition<Q : BaseQueryCriteria<Q#1
   (type parameter of
   net.corda.core.node.services.vault.BaseQueryCriteria.OrComposition)
   , P#1 (type parameter of
   net.corda.core.node.services.vault.BaseQueryCriteria.OrComposition)
   >, P : BaseQueryCriteriaParser<Q#1, P#1>>(a: Q#1, b: Q#1)
2  None of the following substitutions
3  (Q#2 (type parameter of
   net.corda.core.node.services.vault.BaseQueryCriteria),Q#2)
4  (BaseQueryCriteria<Q#2, P#2 (type parameter of
   net.corda.core.node.services.vault.BaseQueryCriteria)>,BaseQueryCri
   teria<Q#2, P#2>)
5  (BaseQueryCriteria<in BaseQueryCriteria<Q#2, P#2>,
   P#2>,BaseQueryCriteria<in BaseQueryCriteria<Q#2, P#2>, P#2>)
```

> Kotlin can quite emulate Scala 🙂

No raw types, so no late generification of types

Many developers rely heavily on type inference to avoid dealing with complex generics cases

What I really really want

Upgrade requests

- **Public API mode**, where omitting types/javadocs/visibility, *Kt classes are errors for public API packages.
- **Java 8 bytecode support** (e.g. default methods)
- **Intentions** for missing @JvmOverloads, @JvmStatic
- Full **Jigsaw** support
- Someone full time on **Dokka** for a while?
- More robust **type inference** (will tolerate some slowness to get this)

Future areas of language risk

- Concepts available for borrowing now nearly exhausted?
- Will Kotlin community fracture like Scala did?
- Kotlin/Java incompatibilities growing as Kotlin accelerates ahead (e.g. modules)
- Community poorly reimplementing OpenJDK (e.g. for /Native and /JS)?

KotlinConf

Gumball

And now for something completely different…

Project goals

- Compile Java/Kotlin apps to **small native executables**
- By embedding a JVM
- Simplified distribution for command line tools.
- Provide some competition for Golang in the command line tools space
- Be easy to use

```
mike@littlegreen ~/S/Gumball> ./gumball --no-proguard --classpath $TESTAPP_CP testapp1.TestApp1Kt testapp
```

Avian JVM

Embedded Java

- Cross platform
- Simple JIT compiler
- Generational GC
- Supports AOT compilation
- Can statically link to native
- Can use OpenJDK library
- Can embed SWT for native GUI

```
void* getIp(MyThread* t, void* ip, void* stack)
{
  // Here we use the convention that, if the return
  // pushed on to the stack automatically as part of
  // stored in the caller's frame, it will be saved
  // instead of on the stack.  See the various imple
  // Assembler::saveFrame for details on how this is
  return t->arch->returnAddressOffset() < 0 ? ip : t
}

void* getIp(MyThread* t)
{
  return getIp(t, t->ip, t->stack);
}

class MyStackWalker : public Processor::StackWalker
 public:
  enum State { Start, Next, Trace, Continuation, Met

  class MyProtector : public Thread::Protector {
   public:
    MyProtector(MyStackWalker* walker) : Protector(w
    {
    }

    virtual void visit(Heap::Visitor* v)
    {
      v->visit(&(walker->method_));
      v->visit(&(walker->target));
      v->visit(&(walker->continuation));
    }

    MyStackWalker* walker;
  };
```

Gumball

Embedded Java for all

- Simple automation utility
- Converts über-JAR to binary object
- Customises JVM bootstrap
- Links to single native image
- Gumball can gumball itself

Limitations and comparisons

Still to-do

- Mac only for now (easy to do Linux/Windows)
- One or two app-compat issues to fix
- ProGuard is really slow
- AOT mode still to do
- One-click option for SWT would be useful

https://github.com/mikehearn/gumball

Comparison

- vs **Kotlin/Native** - Gumball's just Java: no pointers, C interop awkward

- vs **SubstrateVM** (Graal) - similar sized binaries, native-image tool much faster, but SVM is proprietary

Help wanted!

https://github.com/mikehearn/gumball

Thank you!

Mike Hearn

mike@r3.com

mike@plan99.net

https://blog.plan99.net/

#kotlinconf17