REALM

A NEW, EASY TO USE MOBILE DATABASE & OBJECT FRAMEWORK

@DONNFELKER

ABOUT ME

Independent Consultant in mobile and web

Caster.IO - A bite-sized video training platform for Android Developers

I've written a few books on Android.

Have had 2 apps in the top free category on Google Play for over 5 years.

twitter: @donnfelker

WHAT IS REALM?

A FAST, NEW DATABASE WRITTEN FROM THE GROUND UP IN C++

A REPLACEMENT FOR SQLITE

REGULAR JAVA OBJECTS (POJO'S)

```
// Define your model class by extending the RealmObject
public class Dog extends RealmObject {
    @PrimaryKey
    private int id;
    @Required // Name cannot be null
    private String name;
    private int age;


    // ... Generated getters and setters ...
}
```

SAVING OBJECTS

```
// Use them like regular java objects
Dog dog = new Dog();
dog.setName("Rex");
dog.setAge("1");
```

```
// Get a Realm instance
Realm realm = Realm.getDefaultInstance();

// Persist your data easily
realm.beginTransaction();
realm.copyToRealm(dog);
realm.commitTransaction();
```

RETRIEVING DATA

```
Realm realm = Realm.getDefaultInstance();

// Query Realm for all dogs less than 2 years old
RealmResults<Dog> puppies =
            realm.where(Dog.class)
                 .lessThan("age", 2)
                 .findAll();

puppies.size(); // => 1
```

OTHER QUERY MODIFIERS

```
between()
greaterThan()
lessThan()
greaterThanOrEqualTo()
lessThanOrEqualTo()
equalTo()
notEqualTo()
contains()
beginsWith()
endsWith()
```

THANK YOU

HAVE A GOOD DAY

@DONNFELKER

RELATIONSHIPS

```
// Define your relationships with RealmList
public class Person extends RealmObject {

    private String firstName;
    private String lastName;
    private RealmList<Dog> dogs;
    // ... Generated getters and setters ...
}
```

WHERE DO YOU GET IT?

REALM.IO

DOCS AND ALL THAT OTHER GOODNESS

REALM IS FREE

ALL PRODUCTS ARE OPEN SOURCE (ANDROID, IOS, REACT, ETC)

CORE WILL BE OPEN SOURCED - LATER

CORE IS WRITTEN IN C++, FROM THE GROUND UP.

SETUP

```
// Add Realm to the classpath in the root build.gradle file
buildscript {
    repositories {
        jcenter()
    }
    dependencies {
        classpath "io.realm:realm-gradle-plugin:0.88.3"
    }
}

// Then ...

// In your projects build.gradle file
apply plugin: 'realm-android'
```

SETUP CONTINUED

EXTEND REALMOBJECT AND GO!

```
public class Dog extends RealmObject {
    //  ...
}
```

WHY USE IT?

NO MORE SQL

SERIOUSLY THOUGH, YOU GET TO WORK WITH OBJECTS

```
RealmQuery<Dog> query = realm.where(Dog.class);

// Add query conditions:
query.equalTo("name", "Fido");
query.or().equalTo("name", "Odie");

// Execute the query:
RealmResults<Dog> result1 = query.findAll();

// Or do the same all at once (the "Fluent interface"):
RealmResults<Dog> result2 = realm.where(Dog.class)
                                 .equalTo("name", "Fido")
                                 .or()
                                 .equalTo("name", "odie", Case.INSENSITIVE)
                                 .findAll();
```

TRANSACTIONS

```
// Will automatically handle begin/commit, and cancel if an error happens.
realm.executeTransaction(new Realm.Transaction() {
  @Override
  public void execute(Realm realm) {
    Dog dog = realm.where(Dog.class).equalTo("name", "Fido").findFirst();
    dog.setAge(15);
  }
})
```

MANUAL TRANSACTIONS

```
try {
  realm.beginTransaction();
  Dog dog = realm.where(Dog.class).equalTo("name", "Fido").findFirst();
  dog.setAge(15);
  realm.commitTransaction();
} catch (Exception ex) {
  // rollback
  realm.cancelTransaction();
}
```

REACTIVE PROGRAMMING

QUERY RESULTS UPDATE AUTOMATICALLY

#LEMMESHOWYOU

```
Dog d1 = realm.where(Dog.class).equals("id", 123).first();

// register a listener to get notified when the object is updated.
d1.registerChangeListener(new RealmChangeListener() {
    @Override
    public void onChange() { // called once the query complete and on every update
      // do something now that the obj is updated
    }
});

// assume code below is in some other thread/etc  (Android Service, AsyncTask, etc)

// Retrieve the same dog as above
Dog d2 = realm.where(Dog.class).equals("id", 123).first();
realm.beginTransaction();
d2.setAge(12);
realm.commitTransaction();

// d2s change listener gets called after the commit.*
```

REALMRESULTS<T> ARE ALSO AUTO-UPDATING

```
RealmResults<Dog> puppies = realm.where(Dog.class).lessThan("age", 2).first();
puppies.registerChangeListener(new RealmChangeListener() {
    @Override
    // Gets called any time any object that this query represents gets updated
    public void onChange() {
      // do something with the updated results
    }
});

// in some other thread
realm.beginTransaction();
Dog pup = realm.createObject(Dog.class);
pup.setName("Snoop");
pup.setAge(1);
realm.commitTransaction();

// At this piont the puppies change listener will be invoked as the query
// results have automatically been updated.
```

WATCH THE ENTIRE REALM

```
realm.registerChangeListener(new RealmChangeListener() {
    @Override
    // Gets called any time the Realm data changes
    public void onChange() {
      // do something with the updated Realm
    }
});
```

FINE GRAINED

CHANGE LISTENERS ARE COMING

REALM SECURITY

AES-256 ENCRYPTION IS SUPPORTED OUT OF THE BOX

```
// Set up with the config
byte[] key = new byte[64];
new SecureRandom().nextBytes(key);
RealmConfiguration config = new RealmConfiguration.Builder(context)
  .encryptionKey(key)
  .build();

// Realm data is now encrypted
Realm realm = Realm.getInstance(config);
```

MULTI
THREADING

THE ONLY LIMITATION

The only limitation is that you cannot randomly pass Realm objects between threads. If you need the same data on another thread you just need to query for that data on the that other thread. Furthermore, you can observe the changes using Realms reactive architecture. Remember - all objects are kept up to date between threads - Realm will notify you when the data changes.

– Realm Docs

THE GOAL OF REALMS THREADING DECISIONS

The key takeaway here is that Realm makes it effortless to work with data on multiple threads without having to worry about consistency or performance because objects and queries are auto-updating at all times.

– Realm Docs

MULTI THREADING IS HARD

Concurrency in software is difficult [...] Non-trivial multi-threaded programs are incomprehensible to humans.[1]

— Edward A Lee PHD Berkeley University of California

[1] The Problem with Threads PDF Link

HOW ARE THREADING PROBLEMS ARE NORMALLY SOLVED?

LOCKS

Unfortunately, when you dive into the root of the problem you realize you have to lock everything during reads and writes to fully ensure that the data is consistent.


LOCKS ARE
SLOW

THREADING OPTIONS FOR REALM

- Operate on Android's Main Thread[2]
- Use the Async API

[2] Yes, it's possible, but it gives a lot of developers the heebee jeebees.

THE ASYNC API

```
RealmResults<Dog> dogs = realm.where(Dog.class).findAllAsync();
// dogs is an empty list at this point (query is running in the BG)

dogs.addRealmChangeListener(new RealmChangeListener() {
    @Override
    public void onChange() { // called once the query completes and on every update
      // do something with the query results
    }
});

// As soon as the query completes the change listerner will be notified
// that the results are available (and will continue to get notified)
// of new updates

// Working with a single object query
Dog dog = realm.where(Dog.class).equalTo("age", 2).findFirstAsync();
dog.addRealmChangeListener(new RealmChangeListener() {
    @Override
    public void onChange() { // called once the query completes and on every update
      // do something with the dog object
    }
});
```

ANOTHER ASYNC API EXAMPLE

```
RealmResults<Dog> puppies = realm.where(Dog.class).lessThan("age", 2).findAll();
puppies.size(); // => 0 - No puppies in the Realm DB

// Query and update the result asynchronously in another thread
realm.executeTransaction(new Realm.Transaction() {
    @Override
    public void execute(Realm realm) {
        // begin & end transcation calls are done for you.
        Dog theDog = realm.createObject(Dog.class);
        theDog.setAge(3);
        // You could also query and alter objects as well
    }
}, new Realm.Transaction.Callback() {
    @Override
    public void onSuccess() {
      // Original Queries and Realm objects are automatically updated.
      puppies.size(); // => 1 because there is one puppy now
    }
});
```

RXJAVA SUPPORT

```
// Combining Realm, Retrofit and RxJava (Using Retrolambda syntax for brevity)
// Load all persons and merge them with their latest stats from GitHub (if they have any)
Realm realm = Realm.getDefaultInstance();
GitHubService api = retrofit.create(GitHubService.class);
realm.where(Person.class).isNotNull("username").findAllAsync().asObservable()
    .filter(persons.isLoaded)
    .flatMap(persons -> Observable.from(persons))
    .flatMap(person -> api.user(person.getGithubUserName())
.observeOn(AndroidSchedulers.mainThread())
.subscribe(user -> showUser(user));
```

EXAMPLE APP - HTTP://BIT.LY/REALM-RXJAVA

I heard that you can use Realm on the main thread. Why is that possible? Should I?

POSSIBLE? YES.

ADVICE: USE THE ASYNC API

Why is it possible to run on the main thread though?

BECAUSE REALM IS

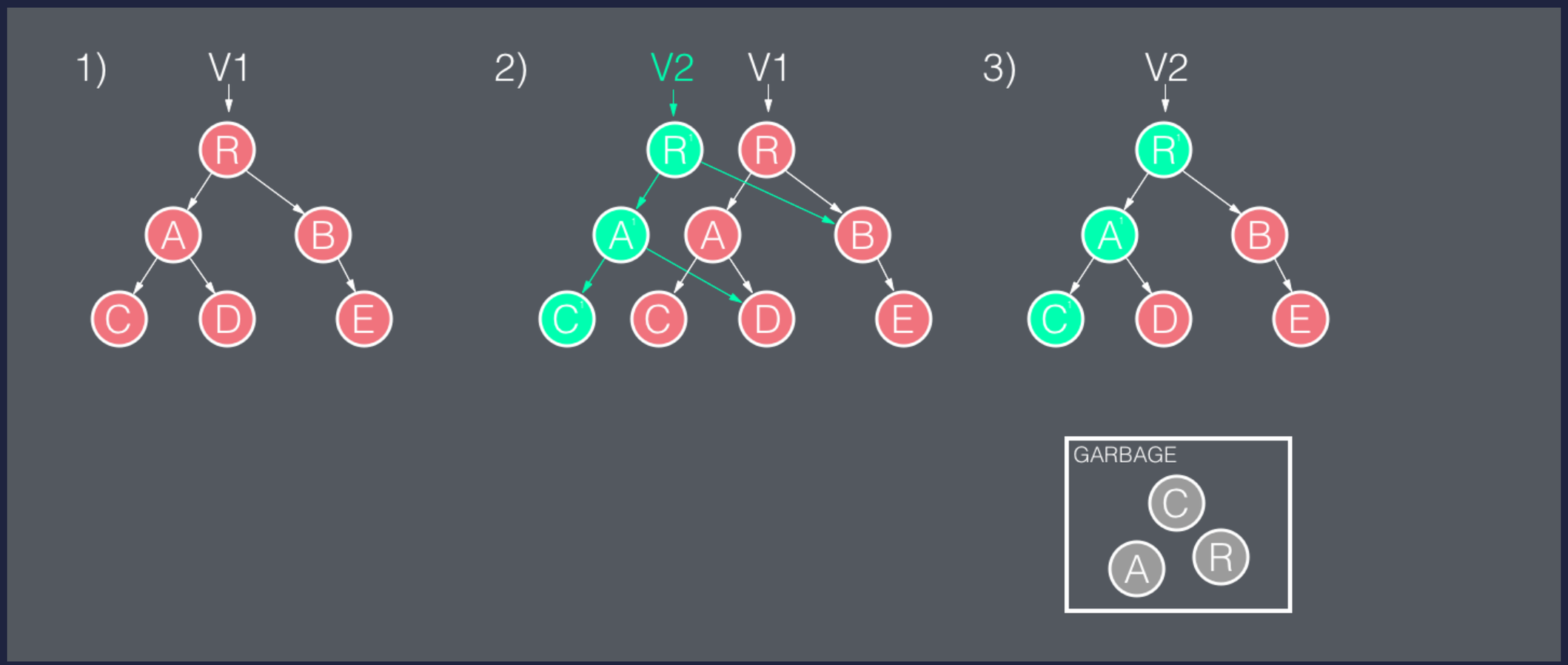
MEGA FAST

WE'LL GET TO THAT IN A SECOND

REALM'S ARCHITECTURE

UNDERSTANDING REALMS INTERNALS

MVCC DATABASE



COPY-ON-WRITE

ZERO-COPY ARCHITECTURE

ORMS MOVE DATA AROUND IN MEMORY

... AND THAT'S OK ...

JUST TAKES TIME AND MEMORY

REALMS DATABASE FILE IS MEMORY MAPPED

The whole file is memory-mapped & is the same format on disk as it is in memory


BUT WHAT DOES THIS EVEN MEAN?

YOU'RE TALKING DIRECTLY TO THE DB AT ALL TIMES, NOT AN ABSTRACTION

CHANGE OF SCHEMA - NO PROBLEM

USE MIGRATIONS

```
// Example migration adding a new class
RealmMigration MyMigration = new RealmMigration() {
  @Override
  public void migrate(DynamicRealm realm, long oldVersion, long newVersion) {

     // DynamicRealm exposes an editable schema
     RealmSchema schema = realm.getSchema();

     // Migrate to version 1: Add a new class.
     // Example:
     // public Person extends RealmObject {
     //     private String name;
     //     private int age;
     //     // getters and setters left out for brevity
     // }
     if (oldVersion == 0) {
        schema.create("Person")
            .addField("id", long.class, FieldAttribute.PRIMARY_KEY)
            .addField("name", String.class)
            .addField("age", int.class);
        oldVersion++;
     }
  }
}


// in your init
RealmConfiguration config = new RealmConfiguration.Builder(context)
    .schemaVersion(1) // Must be bumped when the schema changes
    .migration(new MyMigration()) // Migration to run instead of throwing an exception
    .build()
```

REALM BROWSER

OSX ONLY

default

Search

CLASSES

- Dog 13
- Owner 13

name String	height Float	birthdate Date	vaccinated Boolean	owner <Owner>
Bilbo Fleabaggins	50.000	Sep 8, 2009, 8:26:57 AM	☑	(Tim)
Deputy Dawg	114.000	Jan 26, 2010, 6:58:03 PM	☐	(JP)
Hairy Pawter	114.000	May 13, 2005, 1:36:30 AM	☑	(Arwa)
Jabba the Mutt	98.000	May 16, 2009, 12:41:32 PM	☐	(Joe)
McGruff the Crime Dog	86.000	Sep 14, 2005, 9:38:56 PM	☑	(Alex)
Nerf Herder	44.000	Feb 23, 2001, 5:34:55 AM	☐	(Michael)
Defense Secretary Waggles	63.000	Jun 8, 2006, 6:03:41 AM	☑	(Adam)
Salacious B. Crumb	101.000	Jan 18, 2010, 7:51:19 PM	☑	(Samuel)
Earl Yippington III	44.000	Aug 26, 2001, 9:37:30 PM	☐	(Kristen)
Pickles McPorkchop	71.000	Jan 2, 2008, 6:44:51 PM	☑	(Emily)
Sir Yaps-a-lot	128.000	Jun 18, 2004, 11:26:05 PM	☐	(Katsumi)
Rudy Loosebooty	77.000	Oct 9, 2002, 6:09:45 AM	☑	(Morgan)
Madame Barklouder	78.000	Apr 4, 2009, 9:01:06 PM	☐	(Bjarne)

WHAT'S COMING

REMOVING REQUIREMENT TO EXTEND REALMOBJECT

BETTER RXJAVA SUPPORT FOR CUSTOM SCHEDULERS

MORE PLATFORMS FOR MORE GOODNESS

FEATURE REQUESTS/BUGS/ETC

github.com/realm/realm-java

THE NOT SOO GOOD

NO CUSTOM RXJAVA SCHEDULERS, YET,

NO COMPOSITE PRIMARY KEY CONSTRAINTS, YET.

PARADIGM SHIFT - NO PASSING BETWEEN THREADS.

WHO'S USING THIS ON ANDROID?

A LOT OF COMPANIES. HERE'S A FEW YOU MIGHT HAVE HEARD OF ...

Starbucks, Shyp, Hyatt, IBM, Zappos, Stubhub, Shopsavvy, Virgin Mobile, Subway, Falcon Pro 3, Allegiant Airlines, Digi-Key, Taptalk, Cabify, Karma Wifi ...

WHO'S USING THIS ON OTHER PLATFORMS (IOS)?

JUST LIKE ANDROID - A LOT. HERE'S A FEW YOU MIGHT HAVE HEARD OF ...

Groupon, McDonalds, Zipcar, BBC, ZipRecruiter, Hipmunk, Expensify, Concur, HipChat, Intuit, Oprah, Alibaba, BodyBuilding.com, L'ORÉAL ...

Questions?

THANK YOU

@DONNFELKER