Kotlin:

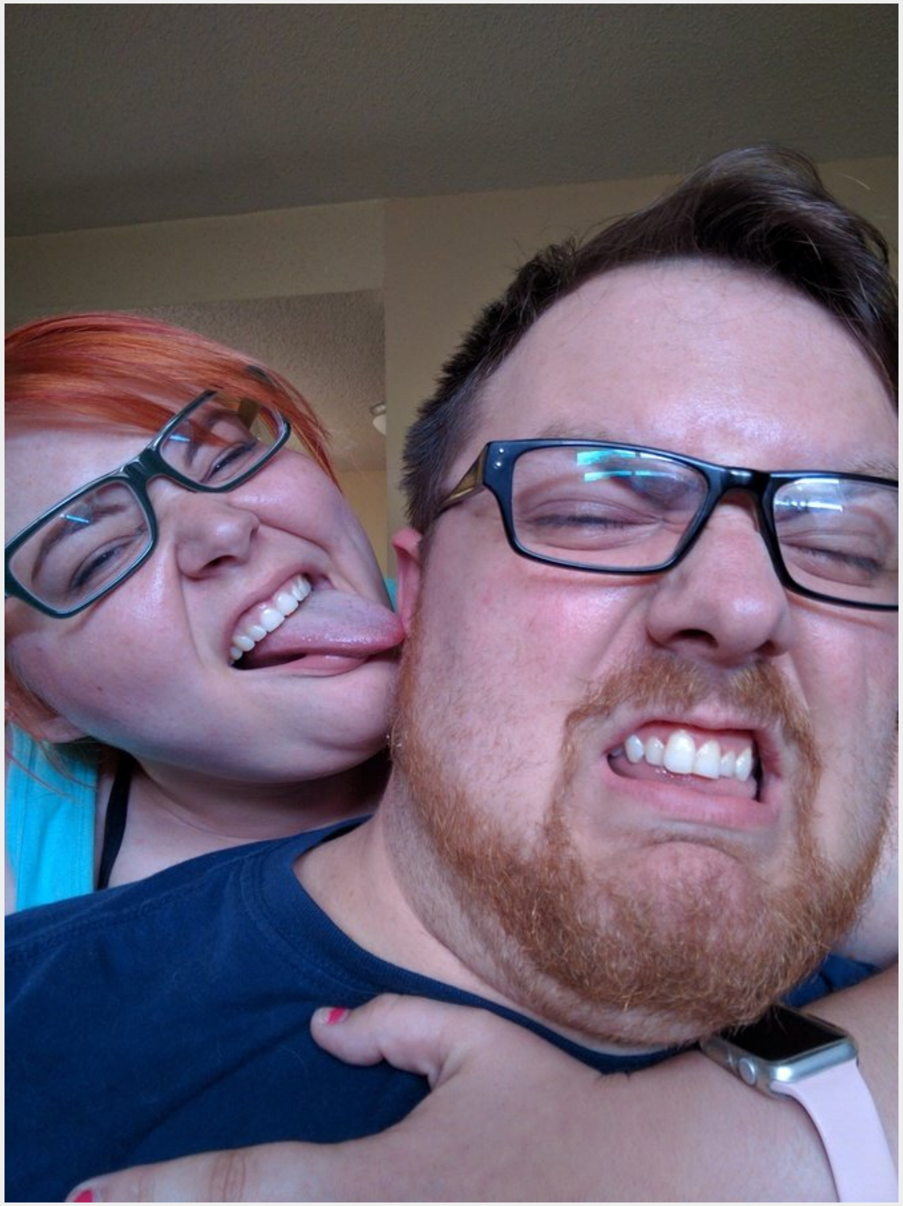
From Russia with Love...ly Syntax

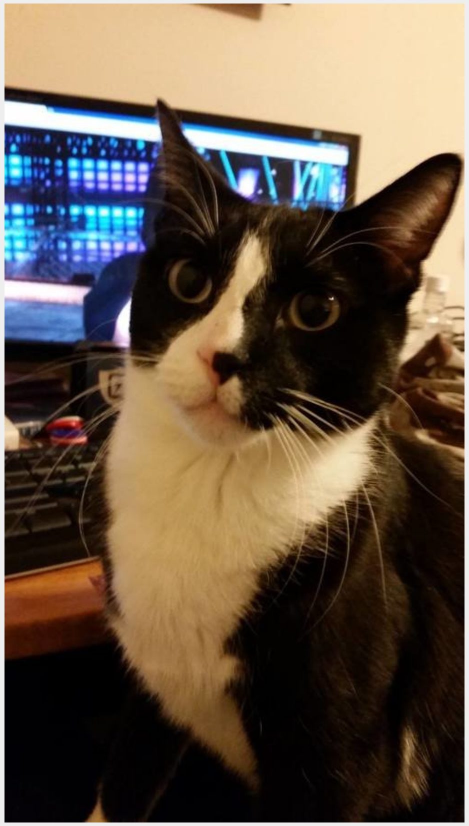
devICT Java Talk

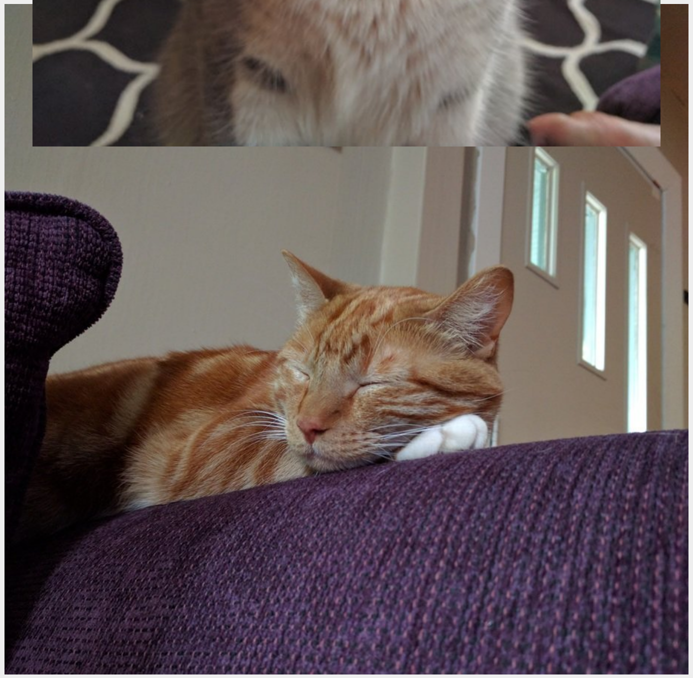
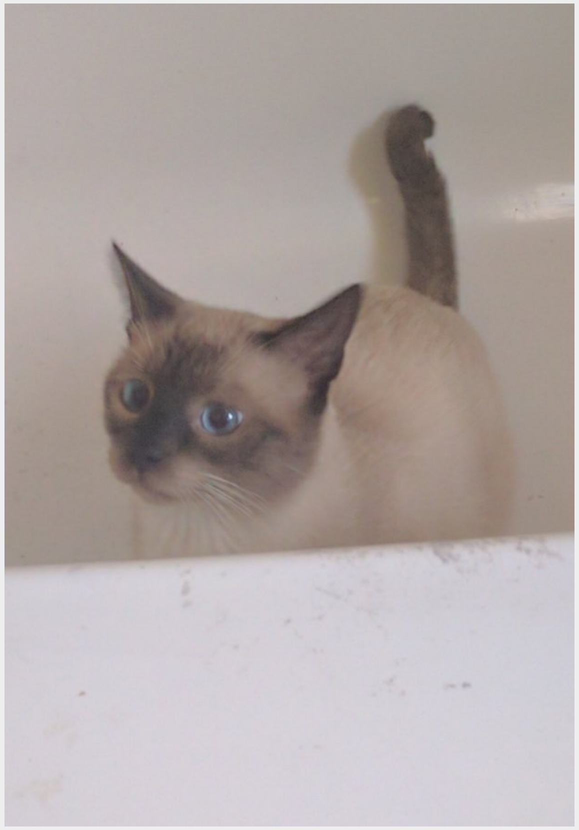
August 16, 2016

Hi.

I'm Jake.





STREET FIGHTER®

MAGIC
The Gathering®

ESPN2

Kotlin

Credit: Anatoly Anatolyevich - 500px.com/photo/146835265/

Project Goals

Pragmatic

Pragmatic

Concise

Pragmatic

Concise

Safe

Pragmatic

Interoperable with Java

Direct Java -> Kotlin Conversion

Targets Java 6

Android Code?

Old Code?

No problem!

First-Class IDE Support

No True Way

This Isn't A Revolution

Concise

Tiny Standard Library

stdlib + runtime ~ 950 kB

```java
public class Dog{
   private final String name;
   private final int age;

   public Dog(String name, int age) {
                this.name = name;
                this.age = age;
        }

   public String getName(){
                return breed;
        }
```

```
class Dog(val name: String, val age: Int)
```

```kotlin
data class Dog(val name: String, val age: Int)
```

HN

REDDIT

```
public void updateWeather(int degreesF) {
        String description;
        Color color; //Color is an enum of a variety of colors

        if (degreesF < 32) {
                description = "freezing";
                color = BLUE;
        } else if (degreesF < 75) {
                description = "mild";
                color = YELLOW;
        } else if (degreesF < 100) {
                description = "hot";
                color = ORANGE;
        } else {
```

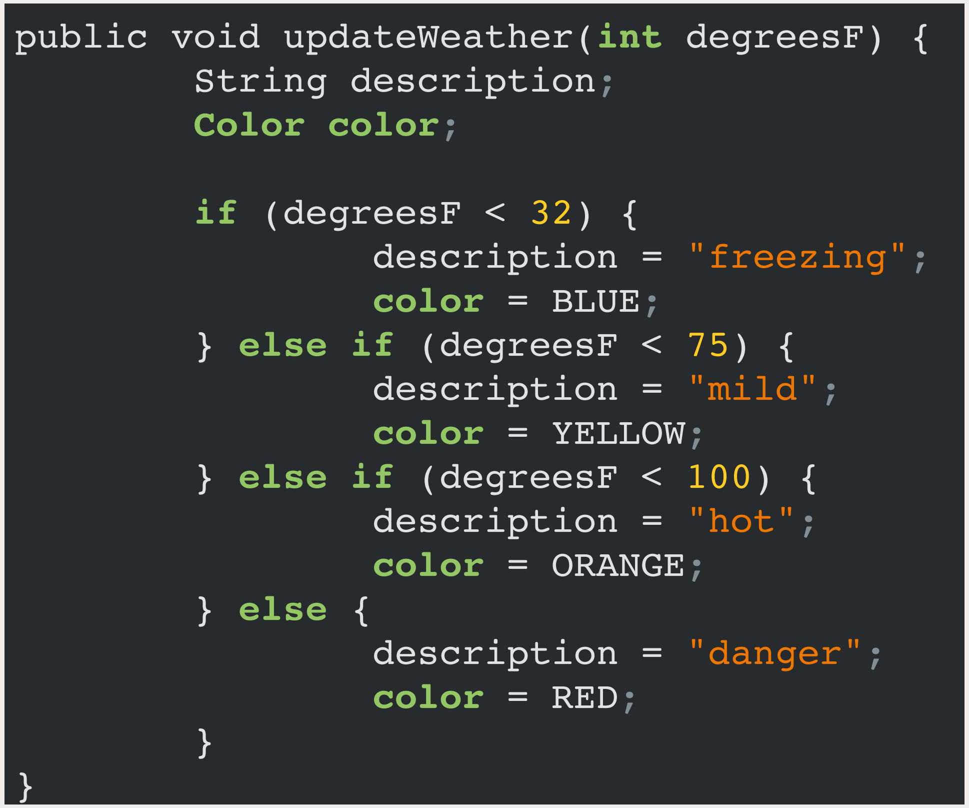
```
fun updateWeather(degreesF: Int) {
    val description: String
    val color: Color

    if (degreesF < 32) {
        description = "freezing"
        color = BLUE
    } else if (degreesF < 75) {
        description = "mild"
        color = YELLOW
    } else if (degreesF < 100) {
        description = "hot"
        color = ORANGE
    } else {
```

```kotlin
fun updateWeather(degreesF: Int) {
    val (description: String, color: Color) =
        if (degreesF < 32) {
            Pair("freezing", BLUE)
        } else if (degreesF < 75) {
            Pair("mild", YELLOW)
        } else if (degreesF < 100) {
            Pair("hot", ORANGE)
        } else {
            Pair("danger", RED)
        }
}
```

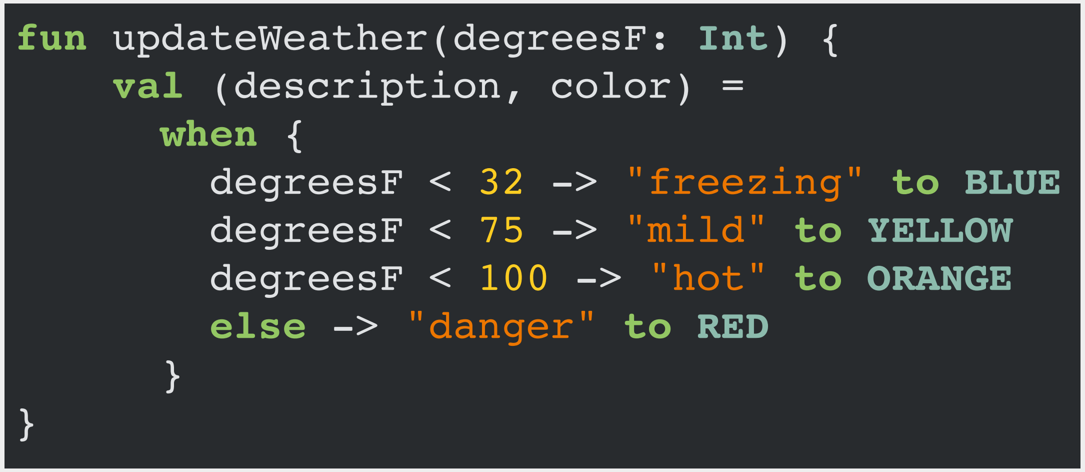
```kotlin
fun updateWeather(degreesF: Int) {
    val (description, color) =
        if (degreesF < 32) {
            Pair("freezing", BLUE)
        } else if (degreesF < 75) {
            Pair("mild", YELLOW)
        } else if (degreesF < 100) {
            Pair("hot", ORANGE)
        } else {
            Pair("danger", RED)
        }
}
```

K3

```kotlin
fun updateWeather(degreesF: Int) {
    val (description, color) =
      when {
        degreesF < 32 -> "freezing" to BLUE
        degreesF < 75 -> "mild" to YELLOW
        degreesF < 100 -> "hot" to ORANGE
        else -> "danger" to RED
      }
}
```



```java
public void updateWeather(int degreesF) {
        String description;
        Color color;

        if (degreesF < 32) {
                description = "freezing";
                color = BLUE;
        } else if (degreesF < 75) {
                description = "mild";
                color = YELLOW;
        } else if (degreesF < 100) {
                description = "hot";
                color = ORANGE;
        } else {
                description = "danger";
                color = RED;
        }
}
```



```kotlin
fun updateWeather(degreesF: Int) {
    val (description, color) =
      when {
        degreesF < 32 -> "freezing" to BLUE
        degreesF < 75 -> "mild" to YELLOW
        degreesF < 100 -> "hot" to ORANGE
        else -> "danger" to RED
      }
}
```

Safety

Null References:
Can't Live With 'Em,
Can't Live Without 'Em

Nullable Types in Kotlin

```
val s1: String = "never null"

val s2: String? = null

s1.length // Will this compile?

s2.length // Will this?
```

Dealing with Nullable Types

```
val s: String?

//Explicitly check
if (s != null) {
    s.length
}

//Safe call operator (returns Int? value)
s?.length

//Assign default value with Elvis Operator
s?.length ?: 0

//Throw an exception on purpose
if (s == null) fail()
s.length
```

Nullable Types
Under the Hood

@Nullable and @NonNullable Annotations

Bits and Bobs

Higher Order Functions

DSLs/Library Support

Coming Soon

Kotlin 1.1 and Beyond

How to Learn Kotlin

- try.kotlinlang.org
  - Basic syntax walkthrough
  - Kotlin Koans
- Kotlin in Action
  - Aimed at Java devs
  - 21 bucks from manning.com with code '39jemerov'
- Udemy - Kotlin Programming: Next Level Java Development
  - Aimed at beginners
  - Only $25 with coupon code 'AMAZINGREADERS25'

Get Involved!

- Kotlin Slack (kotlinslackin.herokuapp.com)
- Contribute! (github.com/JetBrains/Kotlin)
- KEEP (Kotlin Evolution & Enhancement Process)
  - github.com/Kotlin/KEEP

Use Kotlin in Production!

Sources/References

I started to make a slide for this with links and everything.

It got ugly.

So on the repo for this talk,
github.com/jmmoore/javatalk_kotlin,
I've created a sources.txt file with links to videos, talks, and articles I referenced for this presentation.

Thank You!

Email – jacob@kjmoore.us

Twitter – @jmmoore__

GitHub – github.com/jmmoore

Personal Site – jmmoore.tech