

# Kotlin meets Gadsu

---

Christoph Pickl

Kotlin Vienna Meetup – 2017-01-31



- 1 **Introduction to Gadsu**
- 2 **Kotlin in the wild**
- 3 **Code “Schmankerln”**
- 4 **Lessons Learned**

# Introduction to Gadsu

---



This is a cat.



This is Shiatsu . . .



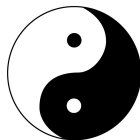
... so is this.



**Gadse**  
+ Shiatsu  
= **Gadsu**



- Acupressure
- Meridian therapy
- Physio therapy
- Massage
- Nervous System stimulation
- Based on the **T**raditional **C**hinese **M**edicine
  - Concept of **Qi** flowing through the body and everything
  - Every aspect of the human grouped into **5 Elements**
  - Body and mind seen as a unit, not separated from each other



Taiji Symbol, Theory of Yin Yang





## Features:

- Client database
- Manage medical records
- Generate reports
- Google integration
- Auto update, auto backup

## Roadmap:


- Pain indicator, 5 Elements
- TCM software agent
- Statistics
- Invoicing
- Doodle integration




- Gradle
- Swing
- Guice
- Spring JDBC
- HSQLDB + Flyway
- Jasper, Pdfbox
- Freemarker
- TestNG, Mockito, Hamcrest
- UISpec4J
- *Initial implementation used Kotlin 0.6 ;)*

Gadsu


Datei Bearbeiten Ansicht Berichte Development




**Anna Nym**  
Behandlungen: 0




**Chuck Norris**  
Behandlungen: 0



**Maxi Mustermann**  
Behandlungen: 3  
Wiedersehen: Mi, 10.08., 20:05



**Pam Anderson**  
Behandlungen: 0



**Queen Liz**  
Behandlungen: 0

**Allgemein** Texte TCM

**Basisdaten**

Vorname

Nachname

Spitzname

Geschlecht

Geburtstag

Sternzeichen

Geburtsort

Herkunft

Beziehungsstatus

Beruf

Kinder

Hobbies

Erstellt am

**Kontaktdaten**

Mail

Telefon

Strasse

PLZ

Stadt

Mi, 10.08., 20:05

Mi, 17.08., 20:05

Neuen Termin erstellen

3. Mittwoch, 27.07.16, 20:00 Uhr

2. Mittwoch, 20.07.16, 20:00 Uhr

1. Mittwoch, 13.07.16, 20:00 Uhr

Neue Behandlung erstellen

**Notiz**

Meine supi wuzi Anmerkung.

Neuen Klienten anlegen Speichern Abbrechen

*Gadsu got something like 35,000 LoC.*

Let's have a look . . .

# Kotlin in the wild

---



```
apply plugin: "kotlin"
buildscript {
    ext.kotlin_version = '1.0.6'
    dependencies {
        classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"
    }
}
dependencies {
    compile "org.jetbrains.kotlin:kotlin-stdlib:$kotlin_version"
    compile "org.jetbrains.kotlin:kotlin-reflect:$kotlin_version"
}
```

Travis CI GmbH

# christophpickl / gadsu

build: passing

Current Branches Build History Pull Requests More options

✓ master minor changes (task tags) -o- #311 passed

- Commit e472ebb
- Compare 9d7a616...e472ebb
- Branch master

⌚ Elapsed time 3 min 3 sec

📅 a day ago

👤 Christoph Pickl authored and committed

Job log View config

Raw log

```
1 Worker information
6 Build system information
73
74 $ export DEBIAN_FRONTEND=noninteractive
118 $ git clone --depth=50 --branch=master
129
130 This job is running on container-based infrastructure, which does not allow use of 'sudo',
```

travis-ci.org



```
language: kotlin
sudo: false
jdk:
  - oraclejdk8
before_install:
  - "chmod +x gradlew"
  - "export DISPLAY=:99.0"
  - "sh -e /etc/init.d/xvfb start"
script:
  - "./gradlew test ..."
notifications:
  email:
    - "MLtravis@gadsu.com"
```



codecov.io

gh : christophickl / gadsu

[Docs](#)
[Support](#)
[Sign up](#)

#87 implemented most of the confirmer logic

christophickl a day ago ✓

9d7a616 master

Diff

Files

Builds

Graphs

Showing 9 of 14 changed files with 56.00% of changed lines covered. [View details](#)

-- / kotlin / at / cpickl / gadsu / mail / module.kt		1	0	0	100%
@@ -12,6 +12,7 @@					
12	12				
13	13	bind(GMailApi::class.java).to(GMailApiImpl::class.java).`in` (Scopes			
14	14	bind(MailSender::class.java).to(MailSenderImpl::class.java).`in` (Sc			
	15 +	bind(AppointmentConfirmationner::class.java).to(AppointmentConfirmat			
15	16				
16	17	bind(MailView::class.java).to(MailSwingView::class.java).`in` (Scope			
17	18	bind(MailController::class.java).asEagerSingleton()			
@@ -18 +19 @@					

codecov.io

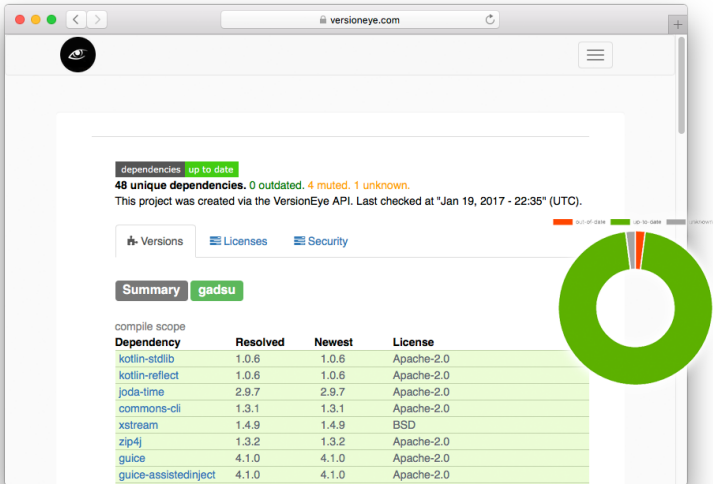


## Gradle Configuration:

```
plugins {  
    id 'jacoco'  
    id 'com.github.kt3k.coveralls'  
}  
jacocoTestReport {  
    reports {  
        xml.enabled = true  
    }  
}}
```

## Travis Configuration:

```
script:  
- "./gradlew ... jacocoTestReport ..."  
after_success:  
- bash <(curl -s https://codecov.io/bash)
```



versioneye.com



## Gradle Configuration:

```
plugins {  
    id "org.standardout.versioneye"  
    version "1.4.0"  
}
```

## Travis Configuration:

```
script:  
- "./gradlew ... versioneye-update ..."
```

# Code Schmankerln

---



The *neutral* **domain object**:

```
package at.cpickl.gadsu.client

data class Client(
    val id: String,
    val name: String
)
```



**Persistence** specific functionality:

```
package at.cpickl.gadsu.persistence

data class ClientDbc(
    val TXT_ID: String,
    val TXT_NAME: String
)

fun Client.toDbc() =
    ClientDbc(id, name)

class ClientRepo {
    fun save(client: Client) {
        saveSomewhere(client.toDbc())
    }
}
```



Add a fluent API to an existing classes:

```
fun <T : JComponent> T.bold(): T {  
    font = font.deriveFont(Font.BOLD)  
    return this  
}  
  
val myLabel = JLabel("text").bold()  
val myTextField = JTextField("text").bold()  
val myTextArea = JTextArea("text").bold()
```



# Extension Properties #1



Possible replacement of common **test factories**:

```
package at.cpickl.gadsu.test

val Client.Companion.testee1: Client
    get() = Client(
        id = "",
        name = "Max Muster"
    )
```

Requires to have some *placeholder*:

```
package at.cpickl.gadsu.client

data class Client( ... ) {
    companion object {}
}
```



Use those testees in your **tests**:

```
package at.cpickl.gadsu.test

@Test class ClientIT {

    @Inject lateinit var repo: ClientRepo

    fun 'reference test scoped testee'() {
        repo.save(
            Client.testee1.copy(name = "Otto")
        )
        // ... assertions ...
    }
}
```

# Lessons Learned

---



- Null handling is a MUST!
- Extension methods for better auto completion
- Named and default arguments, data classes
- Requires developers to be more disciplined
  - Several classes in one (big) file gets common
  - Overuse of single-expression functions
  - Overuse of `apply{}`
  - Explicit type declaration for documentation



- Mostly same as for Java
- Build system support (gradle with kotlin coming!)
- Static code analysis tools missing
- Syntax highlighting mostly missing



- Auto convert from Java to Kotlin
- TODO2
- TODO3
- Automatic replace

```
val text = JTextField()  
val width = text.getWidth()
```

💡 Use property access syntax ▶

# Epilog

---



- Visit the website:  
<https://github.com/christophpickl/gadsu>
- $\text{\LaTeX}$  sources of the slides:  
[https://github.com/christophpickl/gadsu\\_meetup](https://github.com/christophpickl/gadsu_meetup)
- Kotlin-Vienna Usergroup:  
<https://www.meetup.com/Kotlin-Vienna>