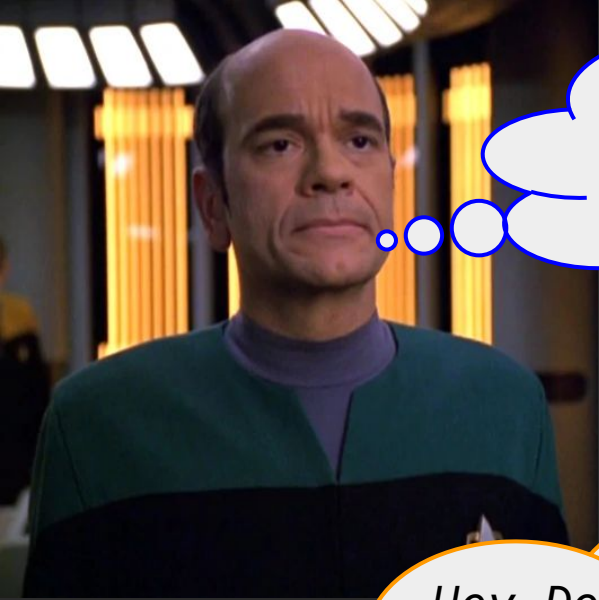




# Koin



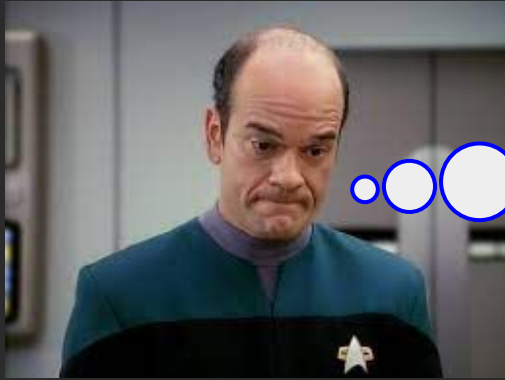
Version 3.0 - Annotations Support



*Please, state the  
nature of the medical  
emergency*

*Hey Doc! There is no  
medical emergency.  
I wanna know how to  
use **KOIN**  
**Annotations.***





Oh, you *Ensign*!  
(sad)  
Great, I will help  
you!



```
buildscript {
```

```
    ext {
```

```
        ksp_version = '1.6.21-1.0.5'
```

```
        koin_ksp_version = '1.0.1'
```

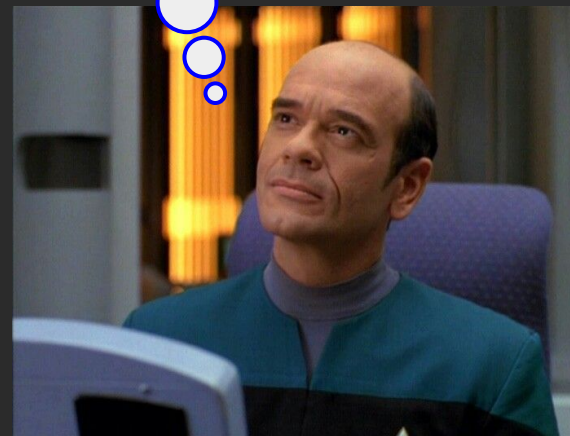
```
        koin_version = '3.2.0'
```

```
        kotlin_version = '1.6.21'
```

```
    }
```

```
}
```

Step 1: Declare your  
dependency versions  
at your **Gradle file**  
in the **project level**.



```
plugins {
```

```
    id 'com.android.application' version '7.2.1' apply false
```

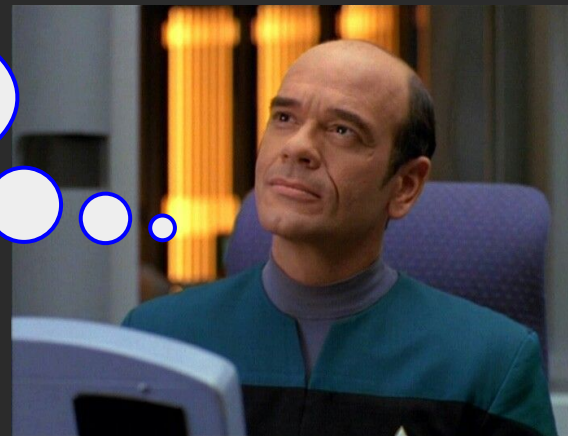
```
    id 'com.android.library' version '7.2.1' apply false
```

```
    id 'org.jetbrains.kotlin.android' version "$kotlin_version" apply false
```

```
    id "com.google.devtools.ksp" version "$ksp_version"
```

```
}
```

Step 2: Specify some configurations because Koin is using ksp.  
Go to your **Gradle file** in the **module level**.



```
plugins {  
    id 'com.android.application'  
    id 'org.jetbrains.kotlin.android'  
  
    id "com.google.devtools.ksp"  
}
```

```
android {  
    compileSdk 32  
  
    defaultConfig {  
        ...  
    }  
  
    buildTypes {  
        ...  
    }  
    compileOptions {  
        ...  
    }  
    kotlinOptions {jvmTarget = '1.8'}  
    buildFeatures {viewBinding true}
```

```
sourceSets.main {  
    java.srcDirs("build/generated/ksp/main/kotlin")  
}
```

*Step 3: At the same  
file, declare your  
dependencies.*



```
implementation "io.insert-koin:koin-core:$koin_version"  
implementation "io.insert-koin:koin-android:$koin_version"  
implementation "io.insert-koin:koin-annotations:$koin_ksp_version"  
ksp "io.insert-koin:koin-ksp-compiler:$koin_ksp_version"
```



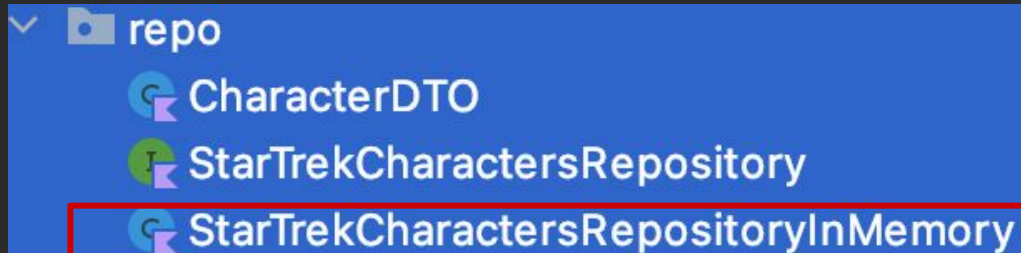
Step 4: Declare your modules. I will show you an example. Don't worry ensign!

**ComponentScan** is a great annotation to map your components:

```
@Module
```

```
@ComponentScan("com.example.koinannotations.repo")
```

```
class RepositoryModule
```

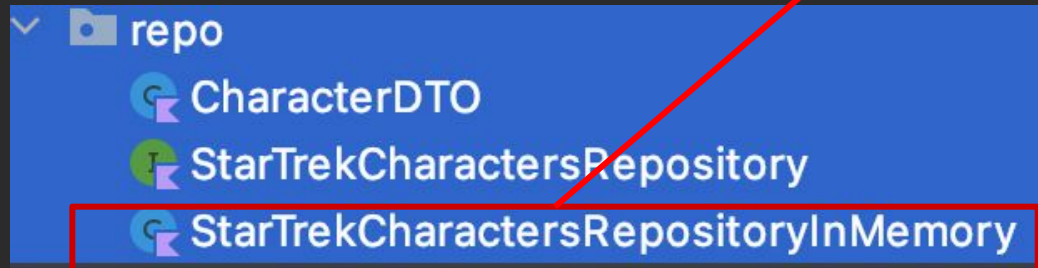


```
@Single(binds = [
    StarTrekCharactersRepository::class
])
class StarTrekCharactersRepositoryInMemory : StarTrekCharactersRepository {
    override fun getCharacters(): List<CharacterDTO> { ... }
```

```
@Module
```

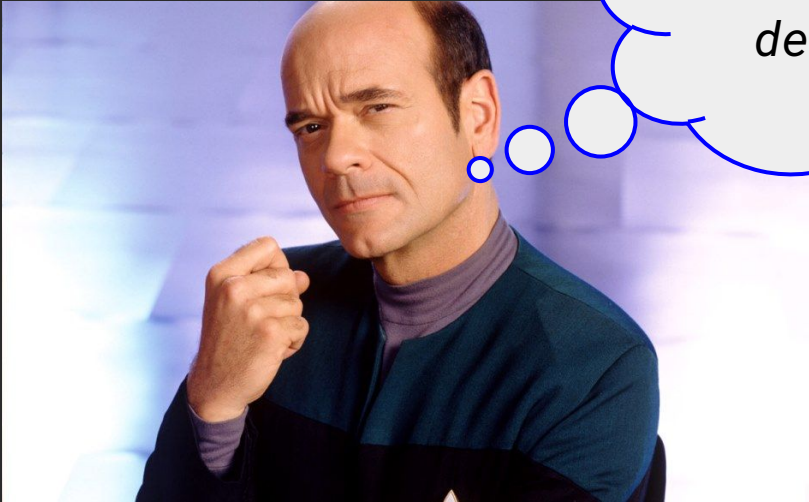
```
@ComponentScan("com.example.koinannotations.repo")
```

```
class RepositoryModule
```



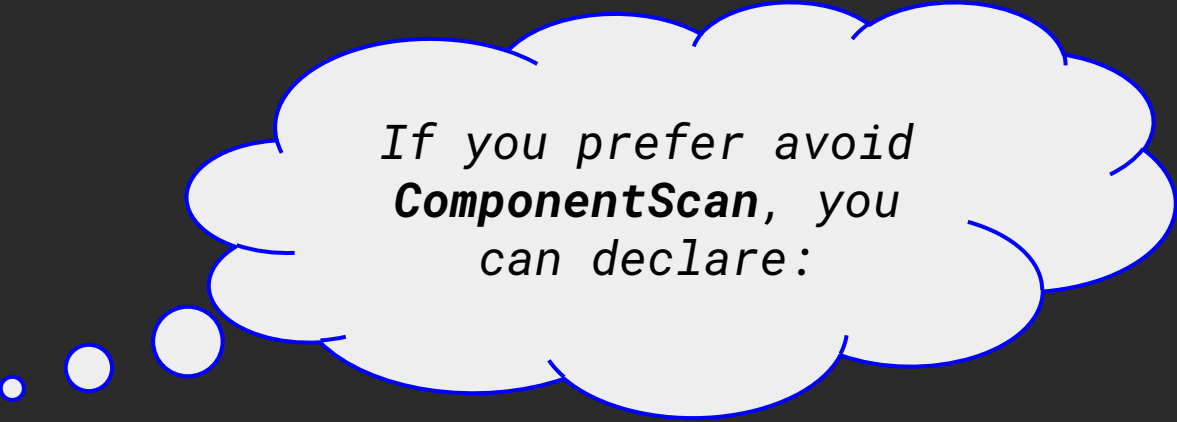
**Single**  
instance  
binding the  
abstraction



A man in a Star Trek uniform, looking thoughtful with his hand on his chin. A thought bubble is connected to his head.

*If your module  
has a  
dependency:*

```
@Module(includes = [RepositoryModule::class])  
@ComponentScan("com.example.koinannotations.domain")  
class DomainModule  
  
@Module  
@ComponentScan("com.example.koinannotations.repo")  
class RepositoryModule
```



*If you prefer avoid  
**ComponentScan**, you  
can declare:*

```
@Module(includes = [DomainModule::class])
class ViewModelsModule {

    @KoinViewModel
    fun mainViewModel(useCase: GetStarTrekCharactersUseCase) : MainViewModel {
        return MainViewModel(useCase)
    }
}
```

```
@Module(includes = [DomainModule::class])  
class ViewModelsModule {
```

```
    @KoinViewModel
```

```
    fun mainViewModel(useCase: GetStarTrekCharactersUseCase) : MainViewModel {  
        return MainViewModel(useCase)  
    }  
}
```



*Required for  
ViewModels...*

```
import org.koin.ksp.generated.module
```

```
class MyApp : Application() {
```

```
    override fun onCreate() {  
        super.onCreate()
```

```
        startKoin { this: KoinApplication
```

```
            modules(  
                
```

```
                ViewModelsModule().module  
            )  
        }  
    }  
}
```

Step 5:  
*Initialize  
your modules...*



```
override fun onCreate() {  
    super.onCreate()  
  
    startKoin { this: KoinApplication  
        modules(  
            ViewModelsModule().module  
        )  
    }  
}
```

***ViewModelsModule** depends  
on the **DomainModule**,  
**DomainModule** depends on  
the **RepositoryModule**.*

*Koin already knows that  
: )*

```
@Module  
@ComponentScan("com.example.koinannotations.repo")  
class RepositoryModule  
  
@Module(includes = [RepositoryModule::class])  
@ComponentScan("com.example.koinannotations.domain")  
class DomainModule  
  
@Module(includes = [DomainModule::class])  
class ViewModelsModule { ... }
```



By Moro