COMPOSE MULTIPLATFORM

STEPHEN.WHITE@CALLISTAENTERPRISE.SE

CADEC 2024.01.18 & 2024.01.24 | CALLISTAENTERPRISE.SE

AGENDA

- What is Compose Multiplatform (CMP)?
- Getting Started
- Demo
- Ongoing Development
- Final thoughts



WHAT IS COMPOSE MULTI PLATFORM ?

WELL, IT'S ALREADY IN YOUR HAND

CALLISTA

CADEC

- CALLISTA DEVELOPER CONFERENCE -

STOCKHOLM 2024.01.18 | GÖTEBORG 2024.01.24

WHAT IS CMP

- It's a Multiplatform UI Framework built by Jetbrains.
- On top of Jetpack compose, a UI toolkit by Google.
- Can be used for Multiplatform development.





Compose Multiplatform

Develop stunning shared UIs for Android, iOS, desktop, and web.



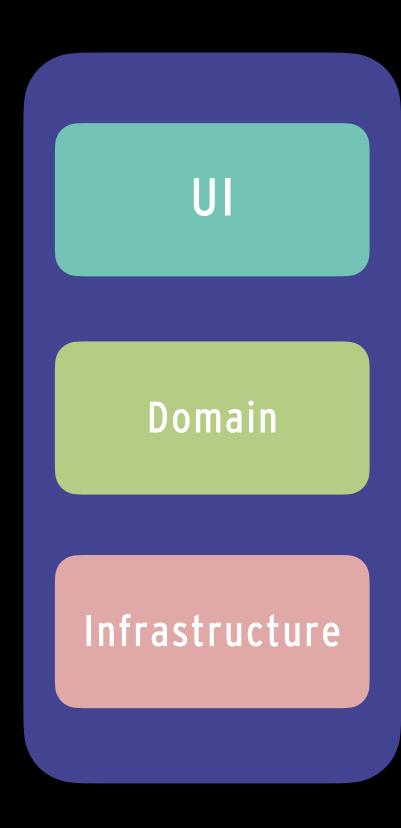
WHAT IS CMP - JETPACK COMPOSE APIS

Jetpack compose APIS

CALLISTA

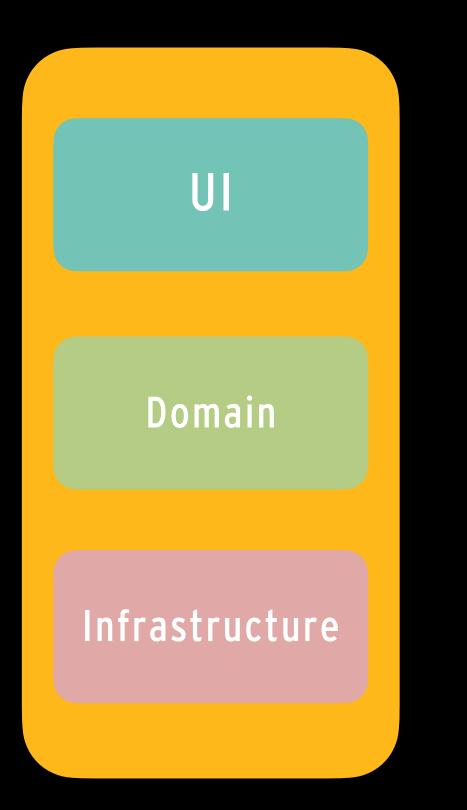
Compose Multiplatform APIs

WHAT IS CMP - TRADITIONAL APPROACH - SEPARATE CODE BASES

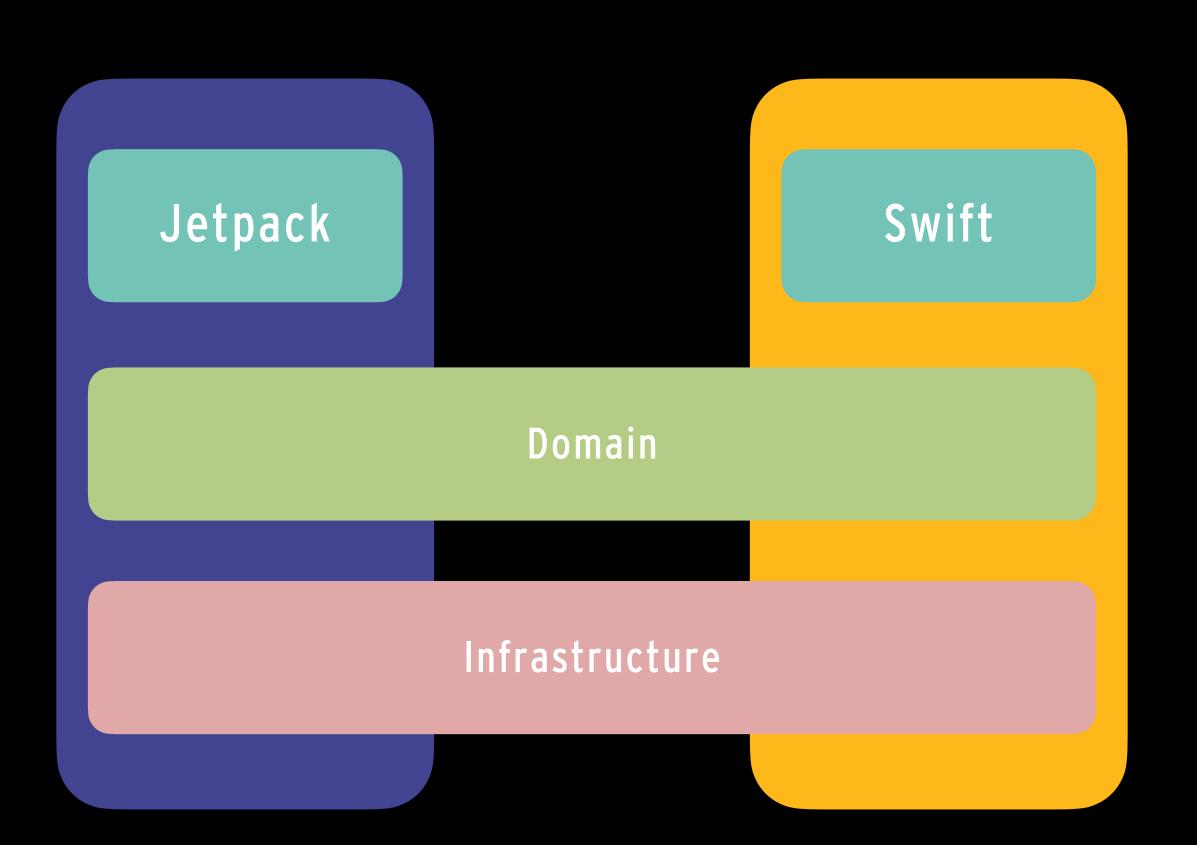




CALLISTA



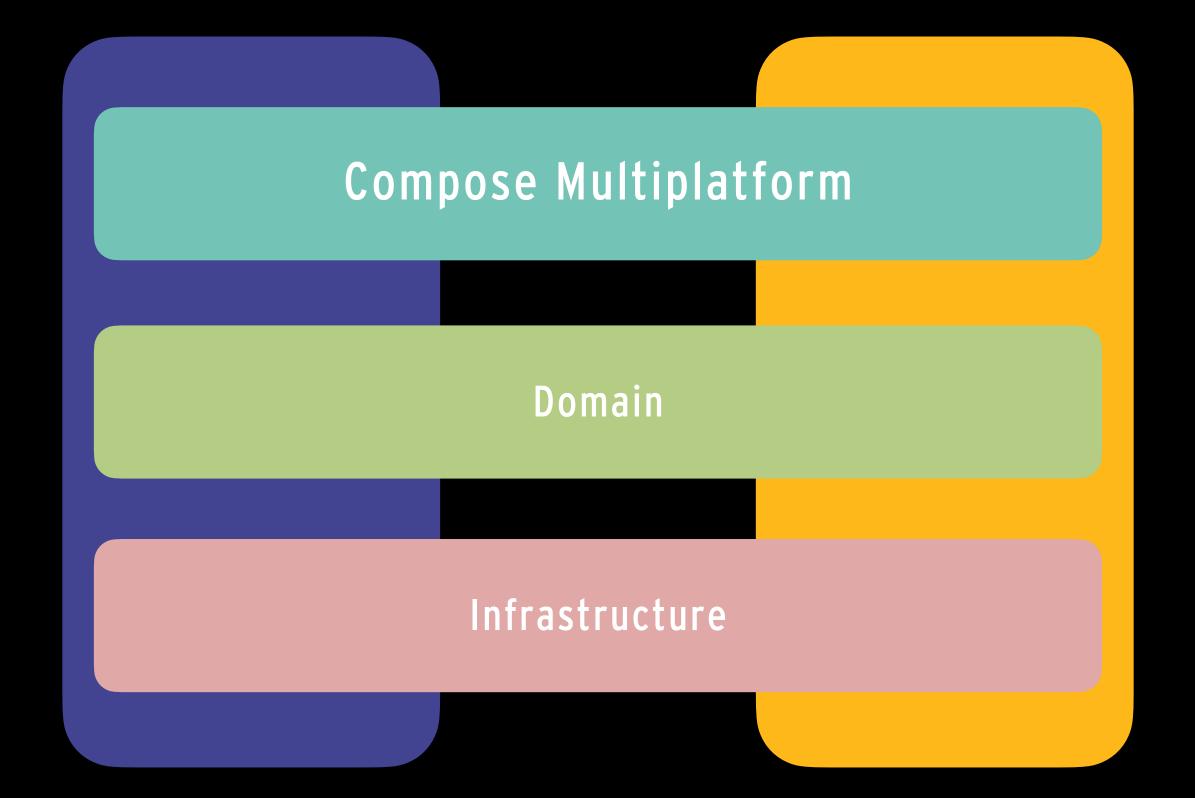
WHAT IS CMP - KOTLIN MULTIPLATFORM







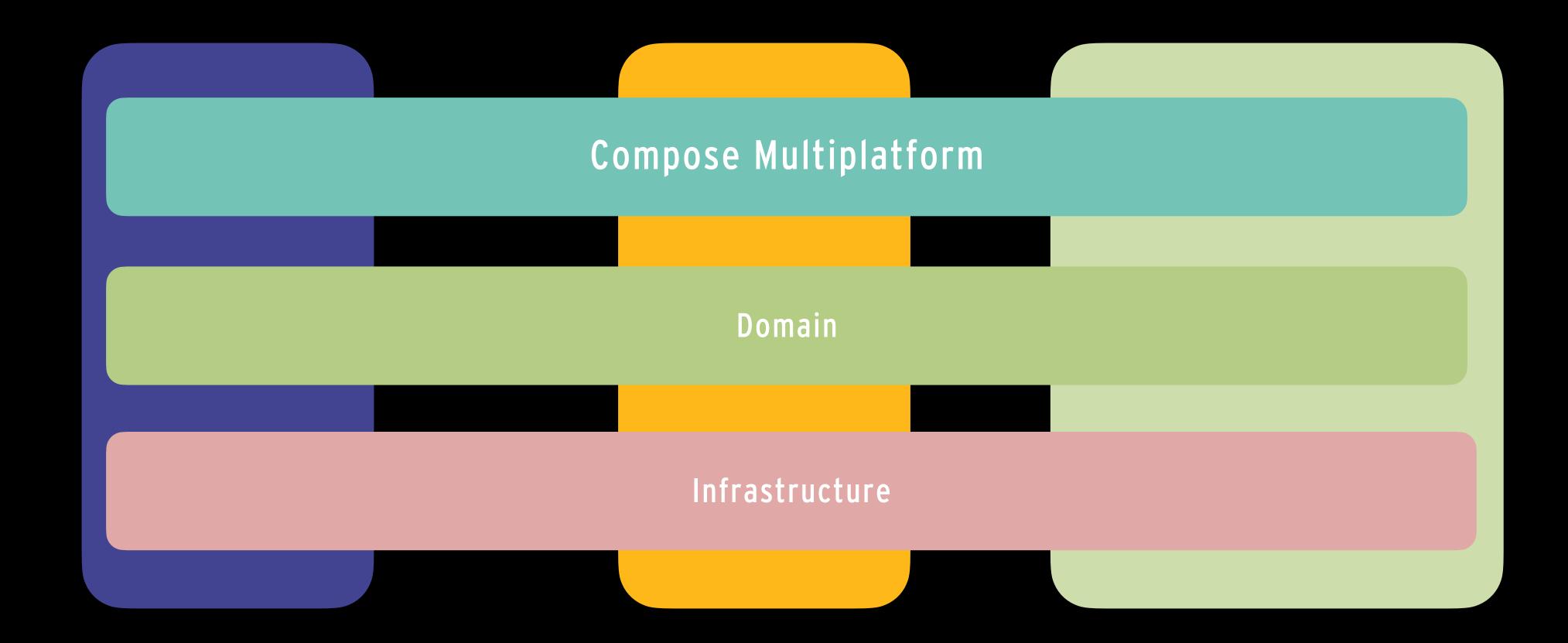
WHAT IS CMP - WITH COMPOSE MULTIPLATFORM







WHAT IS CMP - WITH COMPOSE MULTIPLATFORM... EVEN MORE ...



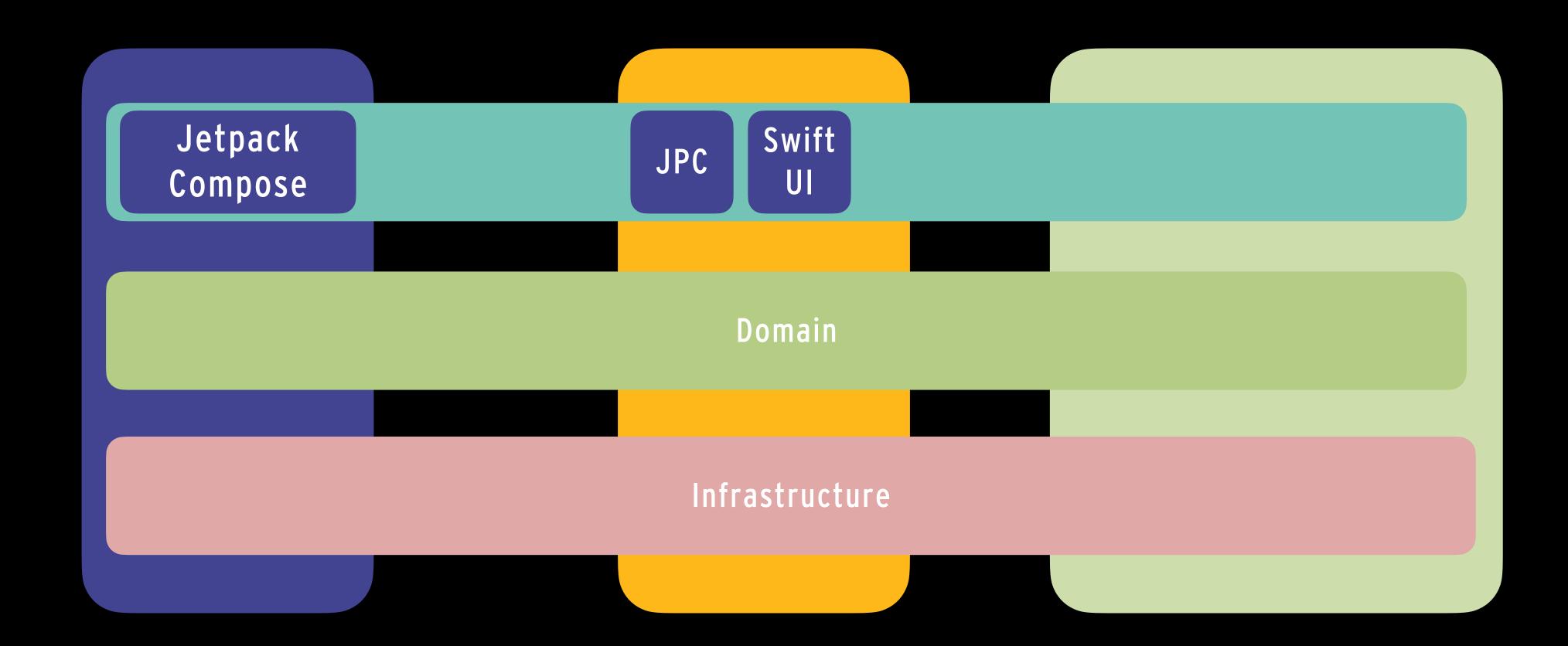
ANDROID

 \mathbf{IOS}

CALLISTA

DESKTOP/WEB

WHAT IS CMP - WITH COMPOSE MULTIPLATFORM



ANDROID

CALLISTA

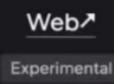
DESKTOP/WEB

WHAT IS CMP - SUPPORTED PLATFORMS

- IOS alpha
- Android via Jetpack Compose
- Desktop Windows, MacOS, Linux
 - Jetbrains toolbox
- Web Experimental (WSM)

	Supported	platforms
ĭOS⊁	0 ☐ Android↗	 Desktop≯
Alpha	via Jetpack Compose	Windows, MacOS, Linux









Services Projects

Installed



Android Studio by Google Hedgehog 2023.1.1



AppCode 2023.1.4



IntelliJ IDEA 2023.3

























WebStorm 2023.3



Fleet 1.28.106 Public Preview

Available



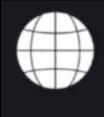


Install

CLion Nova preview, a faster IDE for C and C++

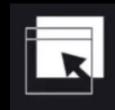


WHAT IS CMP - TIMELINES



Experimental

Web≯



Stable

Desktop >











IOS Beta

WHAT IS CMP - INTEROPERABILITY

- Use Compose Multiplatform inside SwiftUI application
- Use Compose Multiplatform inside UIKit application
- Use UIKit inside Compose Multiplatform
- Use SwiftUI inside Compose Multiplatform
- Its flexible!

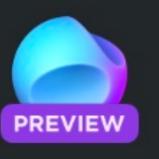


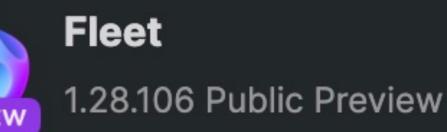


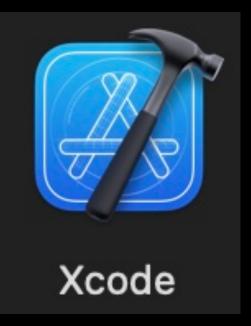
GETTING STARTED - DEVELOPER TOOLS

IDE'S











SIMULATORS



Compose Multiplatform - Kotlin för alla, överallt

Multiplattformsappar, både för webb och mobil, har blivit något av den heliga graalen inom frontend-utveckling. Även om multiplattform kan kännas skrämmande för nativespecialister så lockar möjligheten att ha en enda kodbas som ett generellt botemedel mot ökade kostnader och större komplexitet. React Native och Flutter, som har erbjudit detta under en

tid, har siktat på att kunna erbjuda liknande UI-möjligheter som native-plattformarna och dess språk. Men nu finns

en ny möjlighet - Compose Multiplattform. Med den kan vi använda Kotlin genom alla

lager av applikationen och distribuera den till flera olika frontend-plattformar.

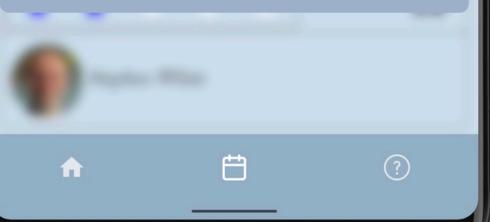
Compose Multiplatform gör det möjligt att dela UI mellan olika plattformar. Det är en

teknologi som befinner sig i stark utveckling som fått stor acceptans från både

utvecklargrupper och de stora teknikleverantörerna. Denna praktiskt inriktade presentation ger en översikt av Compose Multiplatform och dess möjligheter. Vi tittar på hur UI och backend integreras och

hur ekosystemet ser ut.

Slutligen diskuterar vi några för- och nackdelar jämfört med konkurrerande teknologier.



Program Cadec 2024

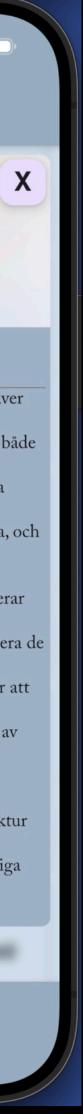
09:24



Design för hög tillgänglighet i en distribuerad arkitektur

Att designa ett system med krav på hög tillgänglighet kräver genomtänkta avväganden gällande hur vi hanterar beroenden mellan komponenter, både inom vårt system och mot befintliga komponenter utanför systemet. Dessa befintliga komponenter kommer vi förmodligen ha ytterst begränsade möjligheter att påverka, och systemdesignen måste därmed göras utifrån de givna förutsättningarna. I denna presentation tittar vi närmare på hur vi kategoriserar olika typer av kopplingar mellan mjukvarukomponenter för att bättre kunna analysera de "tradeoffs" som olika lösningar kommer med, och vilka åtgärder vi kan vidta för att erbjuda bästa tillgänglighet utifrån befintliga begränsningar. Vi diskuterar betydelsen av systemgränser utifrån tillgänglighetsaspekter och hur detta relaterar vidare till organisatoriska aspekter. Vi kommer också att titta på hur normer kring systemarkitektur har förändrats över tid och hur det påverkar vilka lösningar som betraktas som lämpliga för att erbjuda hög tillgänglighet.

Ë



GETTING STARTED - CMP WIZARD

.....

• • < >	kmp.jetbrains.com	5	J (<u>^</u> + C		• • <	> Downloads	
🕽 cadec 🗸 📋 kotlin 🗸 🗎 mongo 🗸	🗀 DOTNET 🗸 📄 Hikes 🗸 📄 kubernetes 🗸	🗎 ddd 🗸 📋 solpaneler 🗸	🗸 🗎 Health 🗸 🗎	∃ vcc ∽	>>>	Name	Size	Kin
📙 cmp-cadec-2024 · Metrie	cs Heroku	Kotlin Multiplatform V	Wizard JetBrains		>	StarBar		- Fol
						StarBar.tar	303 KE	tar
JET BRAINS	Kotlin Multiplatform	h Wizard						
	Rothin Mattiplation							
Ne	w Project Template Gallery Coming soon							
	ject Name							
St	arBar							
	ject ID							
	Android							
	π	_						
w	ith Compose Multiplatform UI framework based on	Jetpack Compose						
	iOS	\checkmark						
U	Implementation							
	Share UI (with Compose Multiplatform UI framew	work) (Alpha)						
(Do not share UI (use only SwiftUI)							
	Desktop							
4	Web Experimental							
	Server							
W	ith Ktor framework							
	DOWNLOAD							

GETTING STARTED - KDOCTOR

maitriyogin/kotlin/cmp—cadec—2024 via 🖝 v20.9.0 via 💎 3.1.0

→ kdoctor

Environment diagnose (to see all details, use -v option):

- [√] Operation System
- [√] Java
- [√] Android Studio
- [√] Xcode
- [√] CocoaPods

Conclusion:

✓ Your operation system is ready for Kotlin Multiplatform Mobile Development!



GETTING STARTED - CODE STRUCTURE

∽ 🕞 StarBar ~/Downloads/StarBar
> 🗋 .fleet
> 🗋 .gradle
> 🗋 .idea
ComposeApp
> 🗅 build
✓ □ src
> 🕞 androidMain [main]
∨ 🗋 commonMain
∨ 🕞 kotlin [commonMain] sources root
🗹 App.kt
C Greeting
🔼 Platform.kt
> 🕞 resources [commonMain] resources root
> 🗋 iosMain
$\mathcal{L}_{\mathbf{C}}^{\mathcal{H}}$ build.gradle.kts

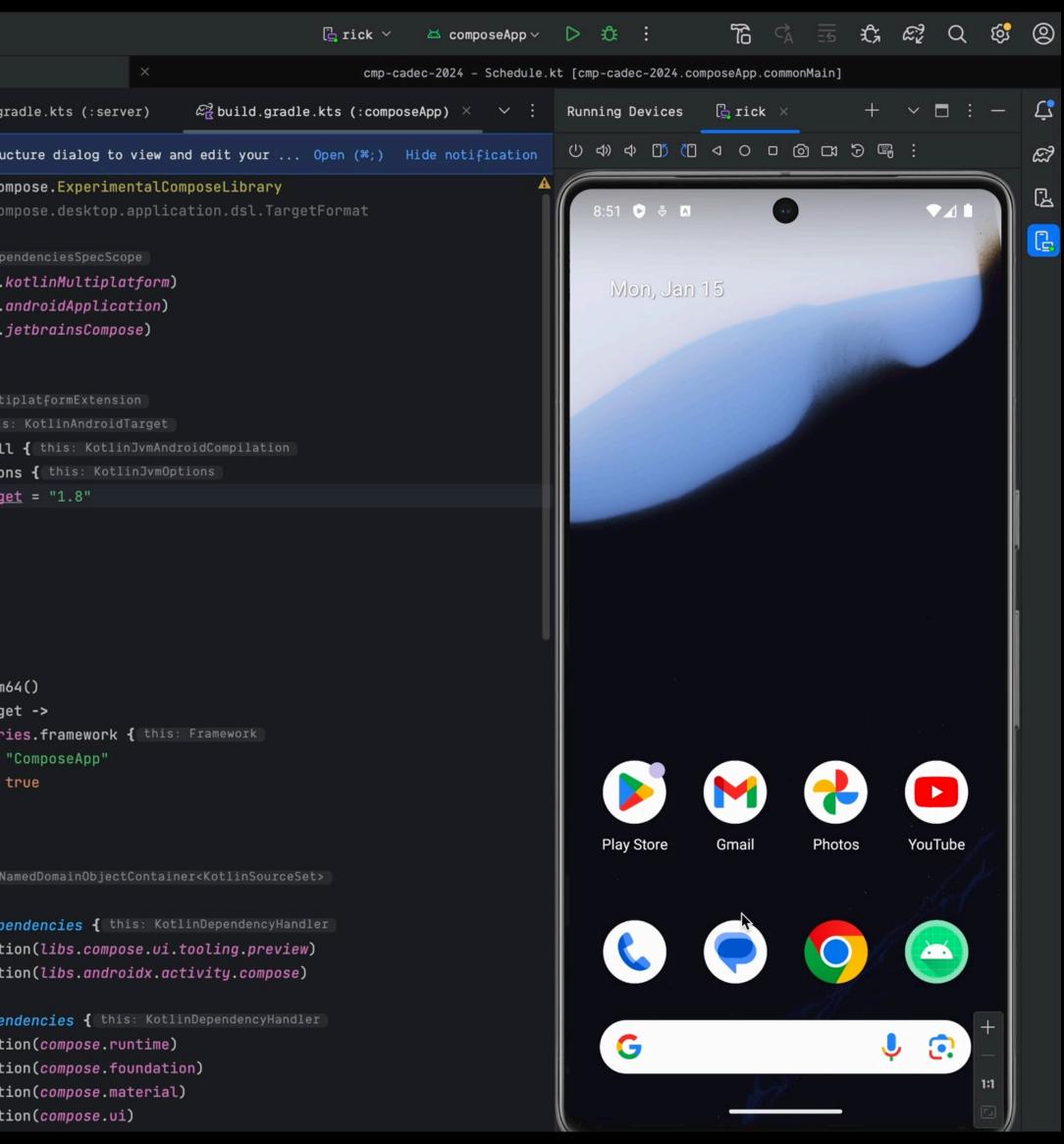
```
@OptIn(ExperimentalResourceApi::class)
@Composable
fun App() {
   MaterialTheme {
       var greetingText : String by remember { mutableStateOf( value: "Hello World!") }
       var showImage : Boolean by remember { mutableStateOf( value: false) }
       Column(Modifier.fillMaxWidth(), horizontalAlignment = Alignment.CenterHorizontally) { this: ColumnS
           Button(onClick = {
               greetingText = "Compose: ${Greeting().greet()}"
               <u>showImage</u> = !<u>showImage</u>
           }) { this: RowScope
               Text(greetingText)
           }
           AnimatedVisibility(showImage) { this: AnimatedVisibilityScope
               Image(
                    painterResource( res: "compose-multiplatform.xml"),
                    contentDescription: null
```



GETTING STARTED - RUN THE APP

Star	Bar 🗸 Version control 🗸				
	StarBar - build.gradle.kts (:co	mposeAp	op)		
П	Project ~	kt	⊠ A	pp.kt	æ ⁹ gbuild.gra
°∆	✓ C StarBar ~/Downloads/StarBar	(î) Y	íou can	use t	ne Project Struc
	> 🗀 .fleet		impor	rt ora	.jetbrains.com
•••	> 🗀 .gradle				.jetbrains.com
	> 🗀 .idea				
	✓ CacomposeApp		plug:	ins {	this: PluginDepe
	> 🗀 build		á	alias(libs.plugins.k
	✓ □ src		6	alias(libs.plugins.a
	> 🗋 androidMain [main]		6	alias(libs.plugins.j
	CommonMain		}		
	✓ □ kotlin [commonMain] sources root				
	☐ App.kt				his: KotlinMulti
	> 🕞 resources [commonMain] resources root		á		dTarget { this:
	> 🗋 iosMain			CO	mpilations.all
	$\mathscr{L}^{\mathfrak{g}}_{\mathfrak{G}}$ build.gradle.kts	13			kotlinOption
	> 🗋 gradle	14 15			jvmTarge }
	> 🗋 iosApp			}	3
	✓ Coserver			}	
	✓ □ src	18			
	✓ C☐ main		1	list0f	(
	<pre> kotlin </pre>				sX64(),
	✓ i sw.test			io	sArm64(),
	Application.kt			io	sSimulatorArm6
	> C resources).forE	ach { iosTarge
				io	sTarget.binari
	$\mathscr{E}^2_{\mathbb{C}}$ build.gradle.kts				<u>baseName</u> = "
	> 🕞 shared				<u>isStatic</u> = t
	Ø.gitignore			}	
	<pre>& Build.gradle.kts</pre>			}	
Г	gradle.properties				Coto (this. No
	D gradlew			source	Sets { this: Na
\bigtriangleup	≡ gradlew.bat			an	droidMain.depe
	<pre> local.properties </pre>				implementati
Ň	M↓ README.md				implementati
(!	$\mathscr{E}^{\mathfrak{P}}_{\mathbb{C}}$ settings.gradle.kts			}	
J	> ff External Libraries				mmonMain.depen
\Leftrightarrow	> ➡ Scratches and Consoles				implementati
					implementati
2					implementati
የዖ					implementati
ΤΫ́					



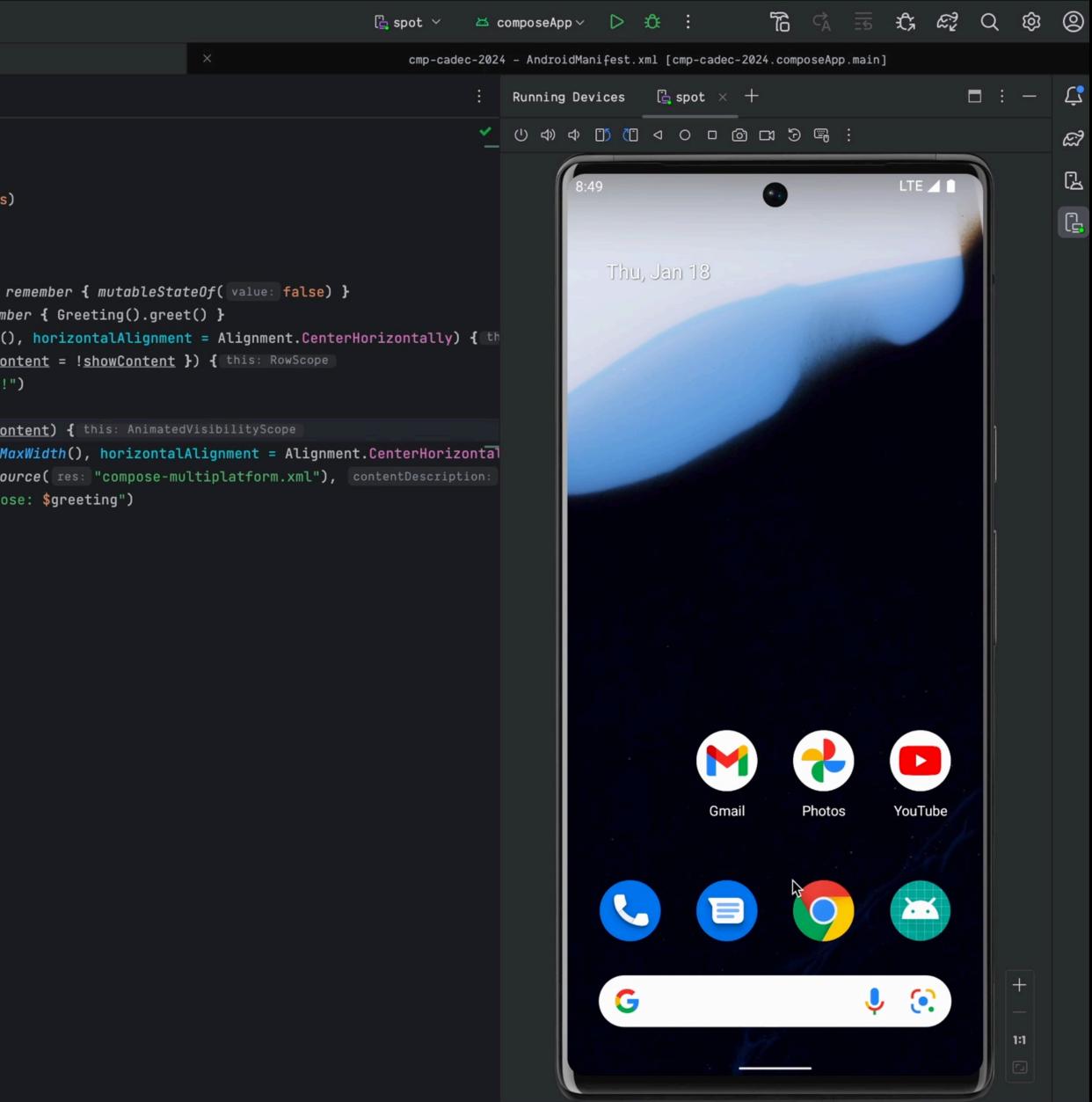


•

GETTING STARTED - OPEN AND RUN THE APP

Star	Bar ∨ ♀ start ∨		
	StarBar - A	.kt	[StarBar.composeApp.commonMain]
	Project ~	🖂 Ap	p.kt ×
-0-	StarBar ~/Downloads/StarBar	1 >	import
	> 🗋 .fleet		
8	> 🗅 .gradle		≗ Stephen White
	> 🗋 .idea		<pre>iOptIn(ExperimentalResourceApi::class)</pre>
	> 🗅 build	19)Composable
	✓ □ composeApp		<pre>Fun App() { MaterialTheme {</pre>
	> 🗅 build		var <u>showContent</u> : Boolean by r
	✓ □ src		val greeting : String = rememb
	> 🕞 androidMain [main]		Column(Modifier.fillMaxWidth()
	∨ 🗀 commonMain	25	Button(onClick = { showCon
	∨ 🕞 kotlin [commonMain] sourc	26	Text(text: "Click me!"
	⊠ App.kt		}
	> Caresources [commonMain] re	28	AnimatedVisibility(showCon
	> 🕞 iosMain	29	Column(Modifier.fillMa
	$\mathscr{E}^{\mathfrak{S}}_{\mathbb{C}}$ build.gradle.kts	30	Image(painterResou
	> 🗋 gradle		Text(text: "Compos }
	> 🗋 iosApp	32 33	}
	∽ 🕞 server		}
	✓ □ src		}
	∨ Camain	36	+
	∨ 🗀 kotlin		
	∨ ⊡ sw.test		
	🖂 Application.kt		
	> 🗋 resources		
Q	$\mathscr{L}^{2}_{\mathbb{C}}$ build.gradle.kts		
	> 🕞 shared		
Ξ	⊘.gitignore		
5	$\mathscr{L}^{\mathfrak{P}}_{\mathbb{K}}$ build.gradle.kts		
T	gradle.properties		
\triangleright	<pre>gradlew</pre>		
	\equiv gradlew.bat		
₽	local.properties		
(!)	M↓ README.md		
\odot	$\mathscr{L}^{?}_{\mathbb{C}}$ settings.gradle.kts		
\Leftrightarrow	> 🗈 External Libraries		
	> ➡ Scratches and Consoles		
▶_			
പ്പ			
	arBar > □composeApp > src > commonMain > (□ katli	$in \geq \Box$ App $kt \geq \widehat{m}$ App
U Sta		a KOCII	LI > C App. Rt > Q App





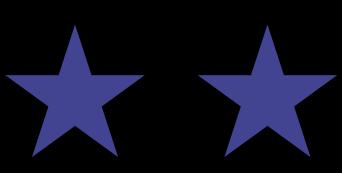












STAR BAR - RECAP

- Look and feel Android
 - Animations
 - Blurs
- Patterns
 - Bottom Sheet
 - Navigation
 - » Stack
 - » Tab
- Resources Images,
 - ▷ Fonts
- Local Storage Multiplatform Settings
- Networking HTTP / WS Ktor client
- Release to app stores ...
- Used it in the Cadec App!!

ONGOING DEVELOPMENT

ONGOING DEVELOPMENT

Material / compose-cupertino

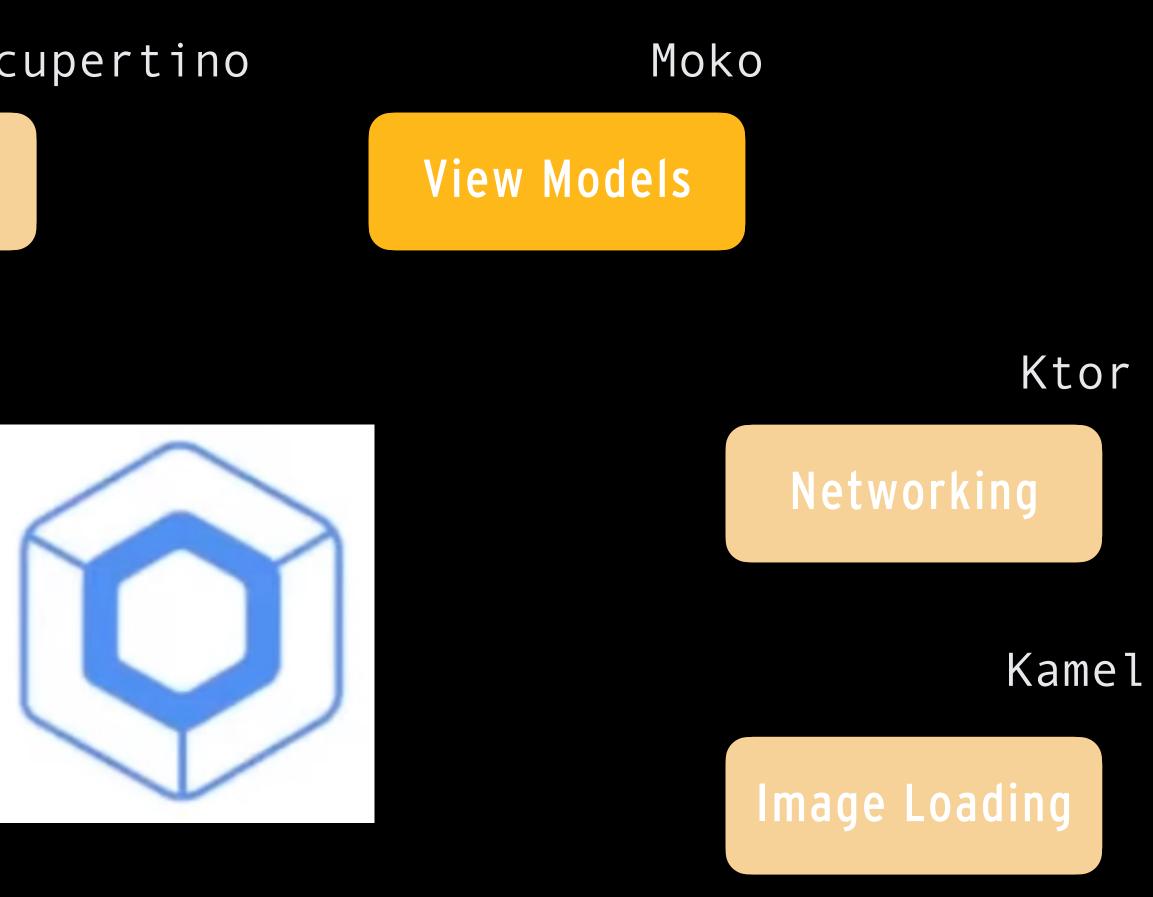
Look and Feel



Resource Management

Voyager

Navigation



SQLDelight / Realm / KStore / Datastore etc ..





Persistence

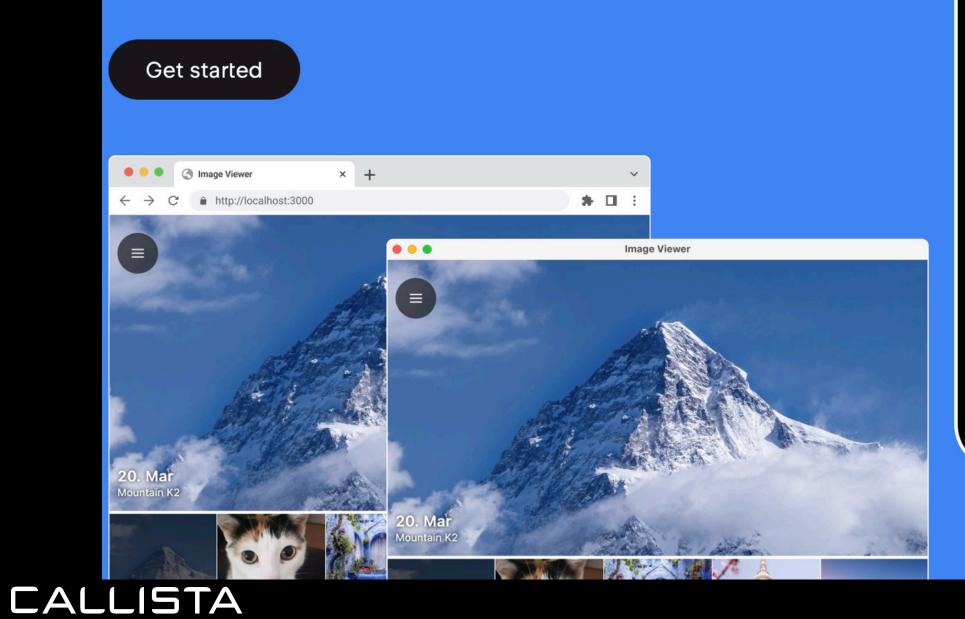
ONGOING DEVELOPMENT - HELP

JET BRAINS



Compose Multiplatform

Develop stunning shared UIs for Android, iOS, desktop, and web.







Note

K2, at 8,611 meters above sea level, is the second-highest mountain on Earth, after Mount Everest.... (see more)

Related memories

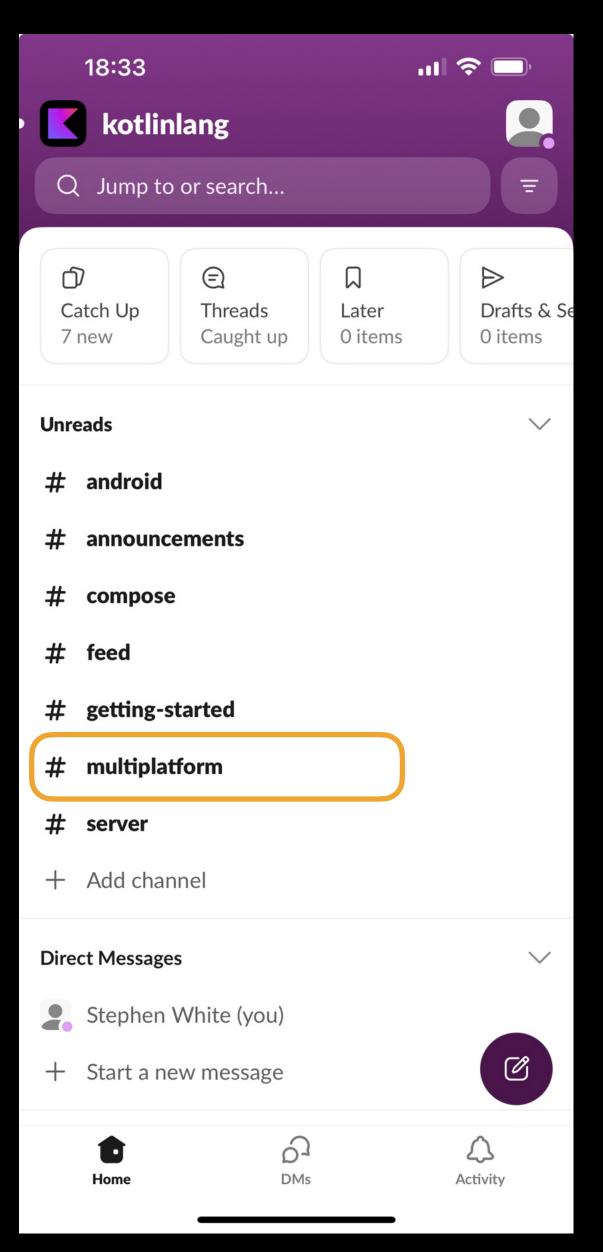


Place









ONGOING DEVELOPMENT - REPLACEMENTS

- Awesome Kotlin Multiplatform
- <u>https://github.com/terrakok/kmp-awesome</u>
- Networking ktor
- Resource Management Mokko
- Navigation Voyager
- Image Loading Kamel
- Look and Feel Material UI
- Persistence -
 - ▷ KStore
 - Realm
 - Multiplatform settings



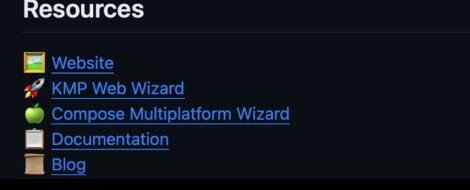
kmp-awesome Public		⊙ Watch 61 →
ᢞ master ╺ ᢪ 1 Branch ा∿ 11 Tags	Q Go to file	t Add file - Code -
😩 mirzemehdi Update README.MD (#82) 🚥		2a42aac · last week 🕚 173 Commits
img	Update readme	4 months ago
🕒 .gitignore	init	2 years ago
README.MD	Update README.MD (#82)	last week
	+	

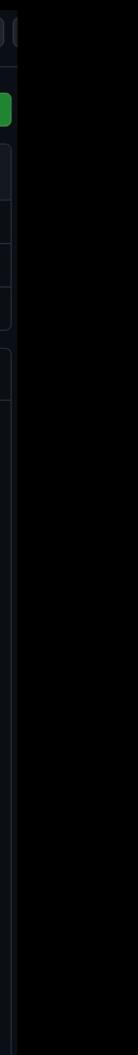
Awesome Kotlin Multiplatform



Kotlin Multiplatform technology simplifies the development of cross-platform projects. It reduces time spent writing and maintaining the same code for different platforms while retaining the flexibility and benefits of native programming

This list contains libraries which support iOS and Android targets in first place.





CALLISTA

FINAL THOUGHTS



TARGET ANDROID TRANSITION TO IOS

EARLY DAYS ...

COMPLEXITY

ACCIDENTAL MULTI PLATFORMS LANGUAGE

CALLISTA

TECH DEBT

ESSENTIAL

DISSONANCE





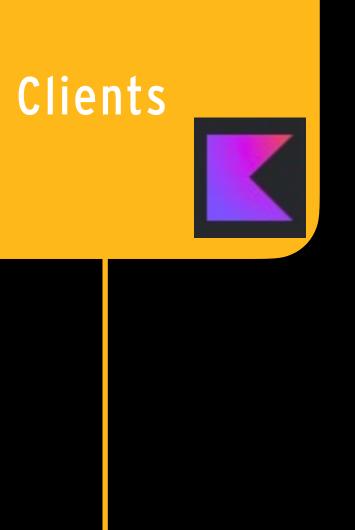














SUMMARY

- Star Bar App
- Cadec App
- Almost production ready!
- Jetpack compose
- Kotlin
- Great community



