

Jornada do QuintoAndar para o Monorepo

Jornada do **QuintoAndar** para o Monorepo

Quem sou eu?

Pedro Zaroni

Senior Software Engineer @ QuintoAndar
Tech Lead @ Native Platform team
Co-organizer @ Flutter Campinas

**Grupo
QuintoAndar**



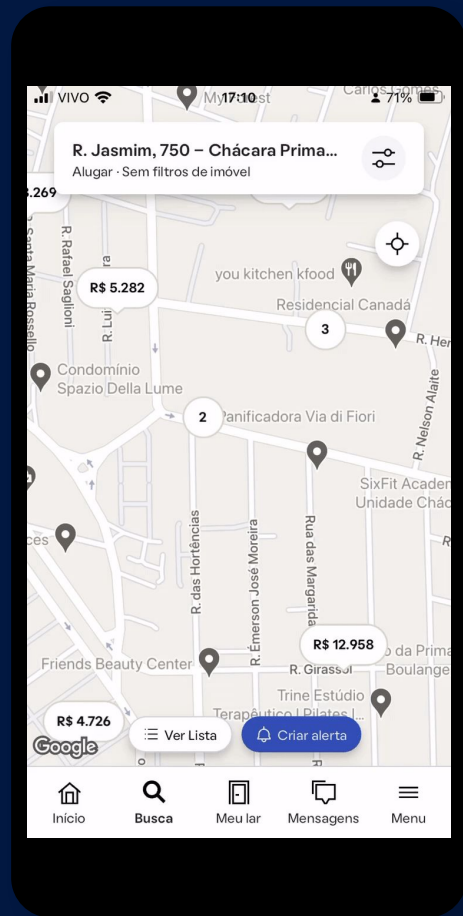
Jornada do **QuintoAndar** para o Monorepo

Sobre o QuintoAndar

A maior plataforma de moradia da
América Latina

Alugamos e vendemos imóveis de
um jeito único no mundo

**Grupo
QuintoAndar**

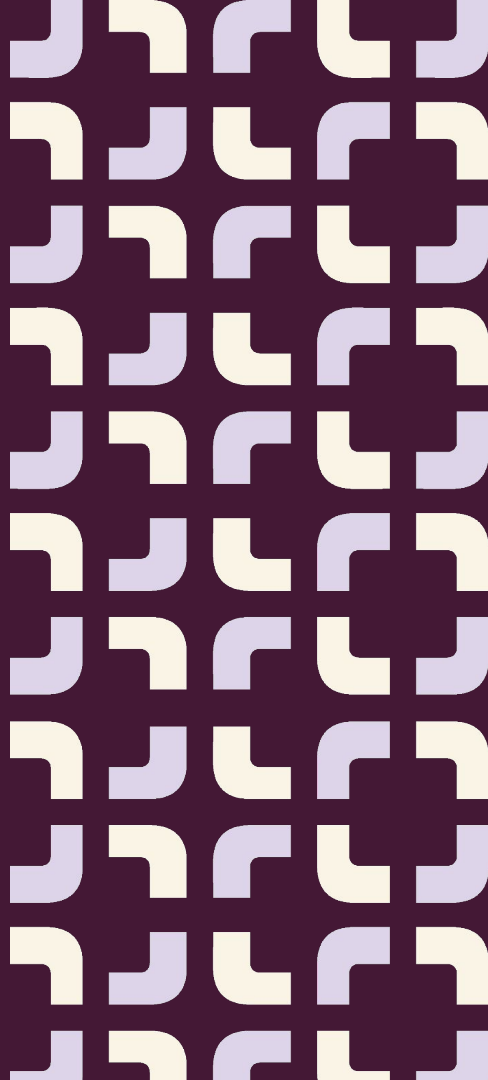


Jornada do **QuintoAndar** para o Monorepo

O que vou falar?

Explorar motivações de escolhermos **unificar todo o código** dart/flutter em um **único repositório**

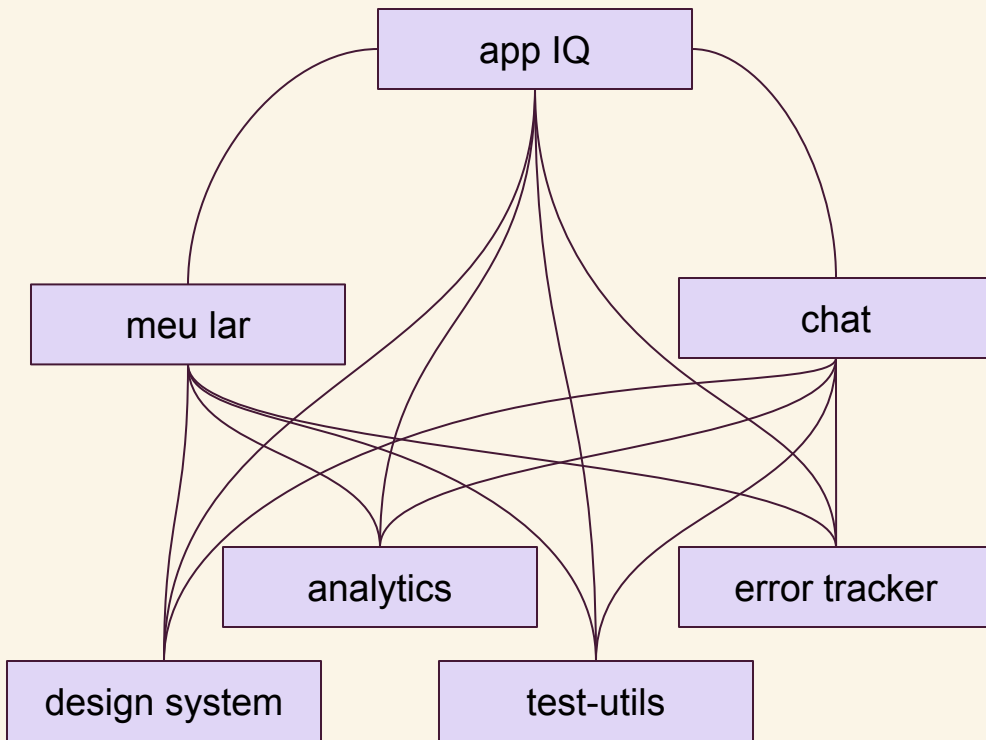
Grupo
QuintoAndar



Antes do monorepo

~7

módulos
formando o
app de IQ



Multi repo

Estratégia modular
(projeto dividido em packages)

Um repositório do GitHub para
cada package Flutter/Dart

quintoandar / **app_inquilino** 🔒

quintoandar / **chat_feature** 🔒

quintoandar / **design_system** 🔒

quintoandar / **analytics** 🔒



```
# pubspec.yaml
name: app_inquilino
description: App de Inquilinos
version: 5.1.0
```

```
# ...
```

```
dependencies:
  design_system: # ...
  analytics: # ...
  meu_lar_feature: # ...
  chat_feature: # ...
```

O que funcionou?



Único dono por repositório

- Código bem auto contido e desacoplado
- Facilidade de definição de CODEOWNERS claros

quintoandar / design_system

```
# .github/CODEOWNERS
```

```
* @quintoandar/design_system_team
```

O que funcionou?



⚡ Tempo para merge rápido

- Necessário executar somente testes, lint, build do package do repositório
- PRs pequenos facilitavam code review



All checks have passed

5 successful checks

O que funcionou?



Rollback de versão

Tão simples quanto alterar a versão no pubspec .yaml raíz

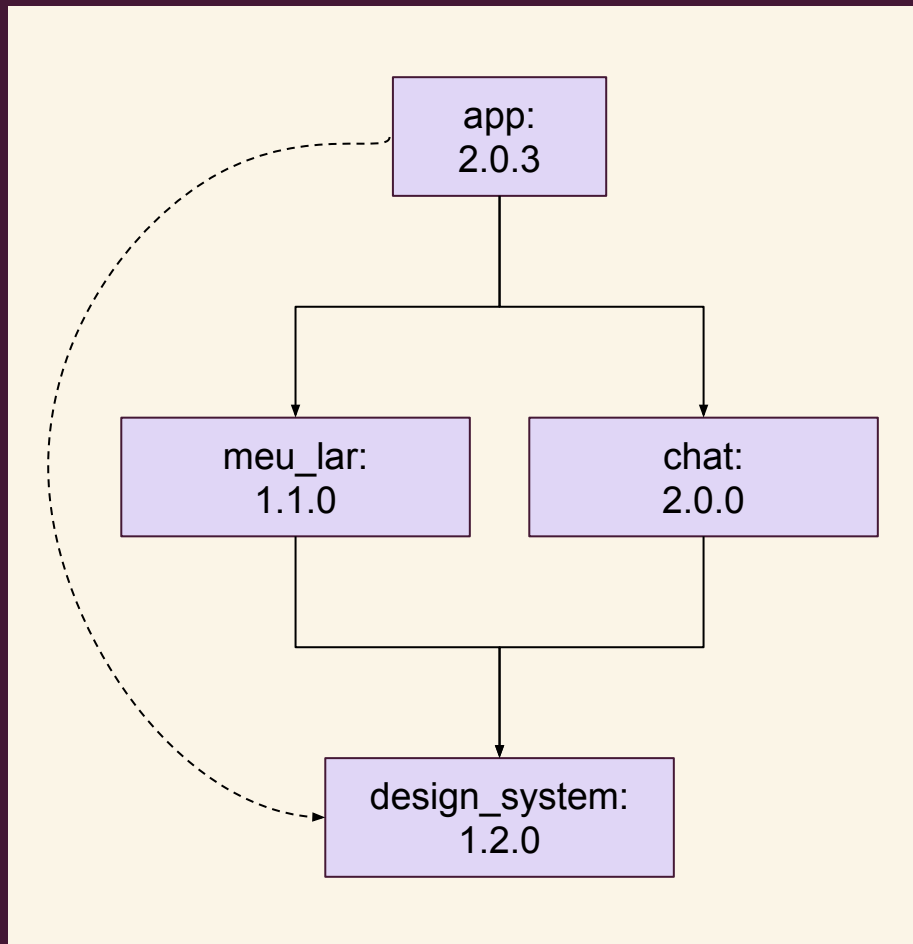


```
dependencies:  
  # ...  
- chat_feature: ^1.3.0 # bug encontrado  
+ chat_feature: ^1.2.5 # última versão estável
```

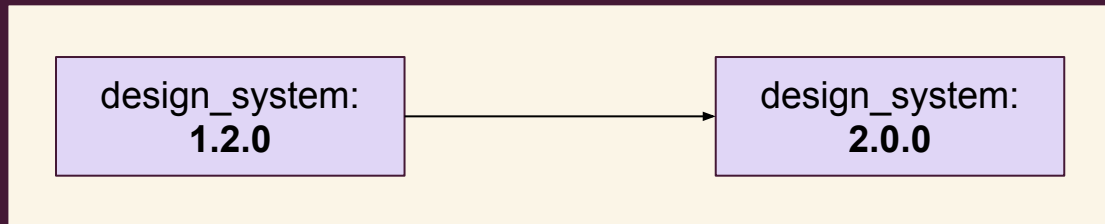
O que não deu tão certo?

⚠ Breaking changes entre packages

- Atualização trabalhosa
- Exige nível de sincronia para evitar bloquear package de receber updates



O que não deu tão certo?

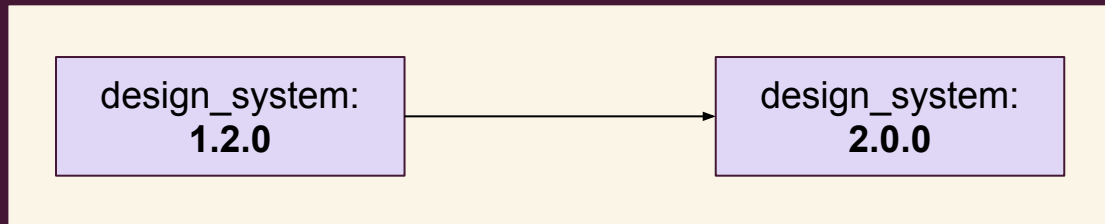


⚠ Breaking changes entre packages

- Atualização trabalhosa
- Exige nível de sincronia para evitar bloquear package de receber updates

```
const Button({  
  required String label,  
  + required ButtonType type,  
  // ...  
});
```

O que não deu tão certo?



⚠ Breaking changes entre packages

- Atualização trabalhosa
- Exige nível de sincronia para evitar bloquear package de receber updates

```
dependencies:  
  # ...  
- intl: ^0.17.0 # flutter 3.7  
+ intl: ^0.18.1 # flutter 3.13
```

O que não deu tão certo?



```
$ flutter pub get
Running "flutter pub get" in app...

Because meu_lar >=1.1.0 <2.0.0
depends on design_system >=1.2.0
<2.0.0
[...]
```

So, because app depends on both meu_lar >=1.1.0 <2.0.0 and chat >=2.0.0 <3.0.0, version solving failed.

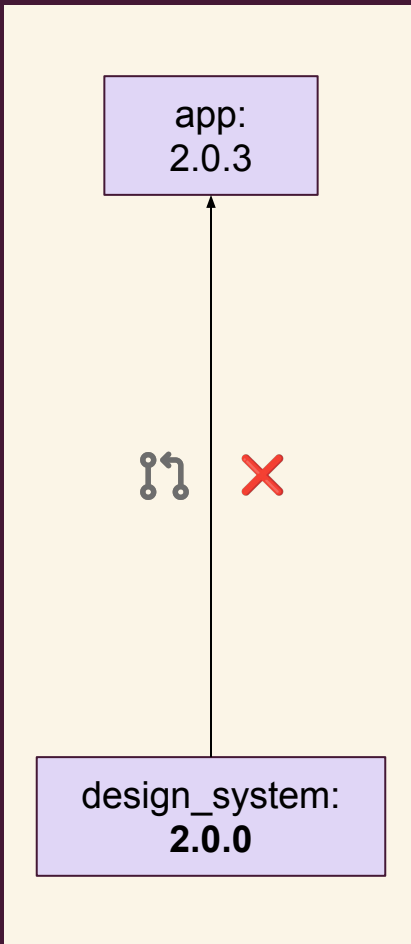
...

exit code 1



Breaking changes entre packages

- Atualização trabalhosa
- Exige nível de sincronia para evitar bloquear package de receber updates

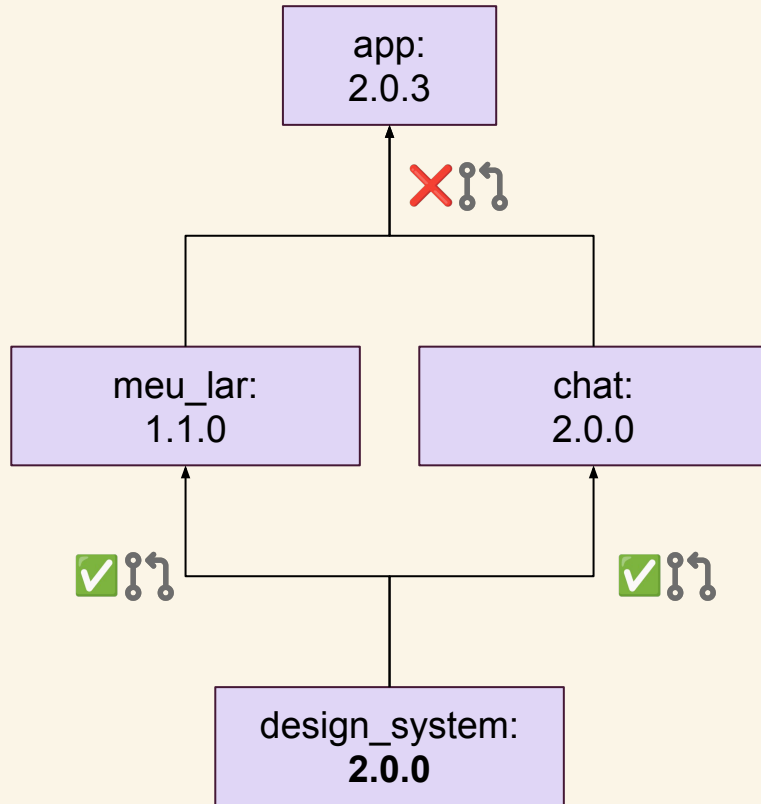


O que não deu tão certo?

🔗 **2** pull requests

⚠️ Breaking changes entre packages

- Atualização trabalhosa
- Exige nível de sincronia para evitar bloquear package de receber updates

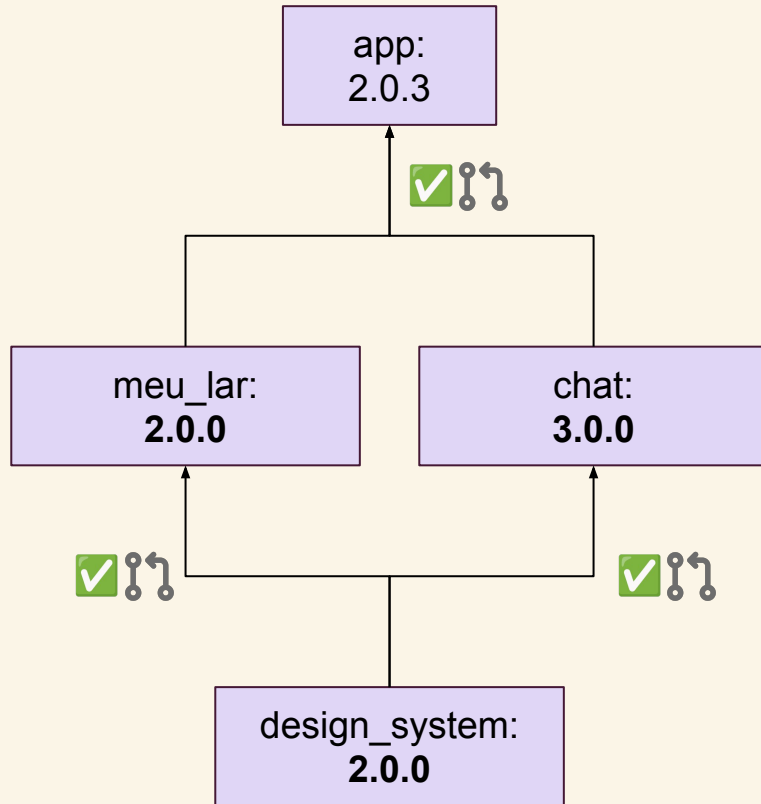


O que não deu tão certo?

🔗 **3** pull requests

⚠️ Breaking changes entre packages

- Atualização trabalhosa
- Exige nível de sincronia para evitar bloquear package de receber updates



O que não deu tão certo?



Muita duplicação de código

- Reutilizar código exigia criar um novo repositório + CI/CD setup
- Times isolados e focados em seus repositórios

quintoandar / app_inquilino 🔒

```
String getWeekdayName(DateTime date) {  
    return DateFormat('EEEE').format(date);  
}
```

quintoandar / chat_feature 🔒

```
String toWeekday(DateTime date) {  
    return DateFormat('EEEE').format(date);  
}
```


O que não deu tão certo?



Ferramental inconsistente

- Cada projeto com uma forma diferente de executar testes, build, etc
- Checks de PR usavam CLI mas nem tudo era automaticamente replicável

quintoandar / app_inquilino 🔒

```
# Makefile
setup:
  flutter pub get
  flutter gen-l10n
  flutter pub run build_runner build --delete-conflicting-outputs
```

quintoandar / chat_feature 🔒

```
# Makefile
build:
  flutter pub get
  flutter pub run build_runner build --delete-conflicting-outputs
```

O que não deu tão certo?

⚠ História de commits desconectada

- Investigar bugs exige maior esforço cognitivo
- Release Notes gerado sem infos detalhadas sobre packages

quintoandar / app_inquilino 🔒

fix: fix deeplink manager (#123) 🗨



mugbug committed 9 hours ago · ✓ 5 / 6

chore: bump design_system to 2.0.0 (#122)



mugbug committed 9 hours ago · ✓ 3 / 3

quintoandar / design_system 🔒

chore(release): release 2.0.0 (#42) 🗨



mugbug committed 9 hours ago · ✓ 1 / 3

feat!: totally new design system layout 🙌 (#41)



mugbug committed 9 hours ago · ✓ 4 / 4


Multi Repo não estava
escalando bem


Aposta em **Monorepo**
para **resolver** maiores
dores e **preparar para**
larga escala

Multi repo

Estratégia modular
(projeto dividido em packages)

Um repositório do GitHub para
cada package Flutter/Dart

quintoandar / **app_inquilino** 

quintoandar / **chat_feature** 

quintoandar / **design_system** 

quintoandar / **analytics** 



```
# pubspec.yaml
name: app_inquilino
description: App de Inquilinos
version: 5.1.0
```

```
# ...
```

```
dependencies:
  design_system: # ...
  analytics: # ...
  meu_lar_feature: # ...
  chat_feature: # ...
```

Monorepo

Estratégia modular
(projeto dividido em packages)

Um repositório do GitHub para
TODOS packages Flutter/Dart

quintoandar / flutter_apps 🔒



```
# pubspec.yaml
name: app_inquilino
description: App de Inquilinos
version: 5.1.0

# ...

dependencies:
  design_system:
    path: packages/core/design_system
  analytics:
    path: packages/core/analytics
  chat:
    path: packages/features/chat
```



Escalou para

70 módulos no monorepo

– sem contar os ainda
não migrados



Mudanças Atômicas entre projetos

- ✓  PR único p/ aplicar alterações que impactam vários packages
- ✓ 😊 Breaking changes deixam de ser um problema
- ✓  Integração das mudanças validada via checks automatizados no PR




feat(flutter): update flutter to 3.13.9 #1000



Merged

mugbug merged 1 commit into `develop` from `feat/flutter-3.13.9`  3 hours ago

Criar novos packages **sem esforço** extra

- ✓  Alto reuso de código entre packages
- ✓  Configurações de CI e comandos locais são reaproveitadas
- ✓  Sem necessidade de lidar com versionamento e publicação de packages em repositório interno

Bloc: New Bloc



Bloc: New Cubit

Mason: Make Global Brick

Mason: Make Local Brick

Mason: New Brick

Maior consistência em ferramentas e comandos

- ✓  Maior facilidade de engenheiros contribuírem para diversos packages
- ✓  Simplicidade de manutenção e evolução de pipelines e ferramentas



```
$ melos run setup  
  
$ melos run test:all
```

Mas...
outros
problemas
apareceram



Quantidade de packages crescendo sem controle



Paved Road

Evolução da documentação, com recomendações para criação, organização e integração de packages



Guardrails

Definimos novos requisitos para criação de packages

- RFC
 - Code owner
-



Flutter Champions

Selecionamos um grupo de Engenheiros mais experientes para garantir a evolução saudável da codebase via Code Review, suporte via Q&A interno, etc



Tempo de merge de PRs decolou

5min -> **50**min



Merge Queue

Habilitamos feature do GitHub para mitigar impacto de atualização com branch principal



Otimização de pipelines de CI

- Removemos/diminuímos alguns usos de code gen como Injectable e ajustamos regras no `build.yaml`
 - `very_good test --optimization`
-



Notificação de PRs pendentes

Criamos Bot que envia lista de PRs pendentes de revisão para o canal do respectivo owner



Injeção de dependência com **service locators** se tornou **inviável** de manter



Injeção via construtor

- Dependências dos packages se tornam mais **explícitas**
- **Erros em tempo de compilação** quando falta alguma dependência



Composition Root

- Padrão para **diminuir boilerplate** por conta de injeção via construtor
- Evita sempre precisar saber dependências da classe que queremos instanciar
- Local centralizado de construção de objetos



```
return myCompositionRoot.makeMyScreen();
```

Ainda temos muito a evoluir



- ✓ ⚡ **Otimização de pipelines CI/CD**
 - Executar lint, build, testes apenas para packages afetados em PRs
 - Abusar do paralelismo que a estrutura modular habilita
 - **Cache** local e distribuído

- ✓ 📦 **Migrar restante dos packages**

Ainda temos alguns packages que serão desafiadores, como o de Design System, que em muitos casos geraria um PR imenso de alterações

Ainda temos muito a **evoluir** e **experimental**

-
- ✓ ⚡ **Otimização de pipelines CI/CD**
 - Executar lint, build, testes apenas para packages afetados em PRs
 - Abusar do paralelismo que a estrutura modular habilita
 - **Cache** local e distribuído
-
- ✓ 📦 **Migrar restante dos packages**

Ainda temos alguns packages que serão desafiadores, como o de Design System, que em muitos casos geraria um PR imenso de alterações
-
- ✓ 🌐 **Novos apps – Flutter Web**

Estamos **experimentando Flutter Web** para alguns projetos que estarão dentro do Monorepo
+ deploys para gerenciar



Grupo QuintoAndar