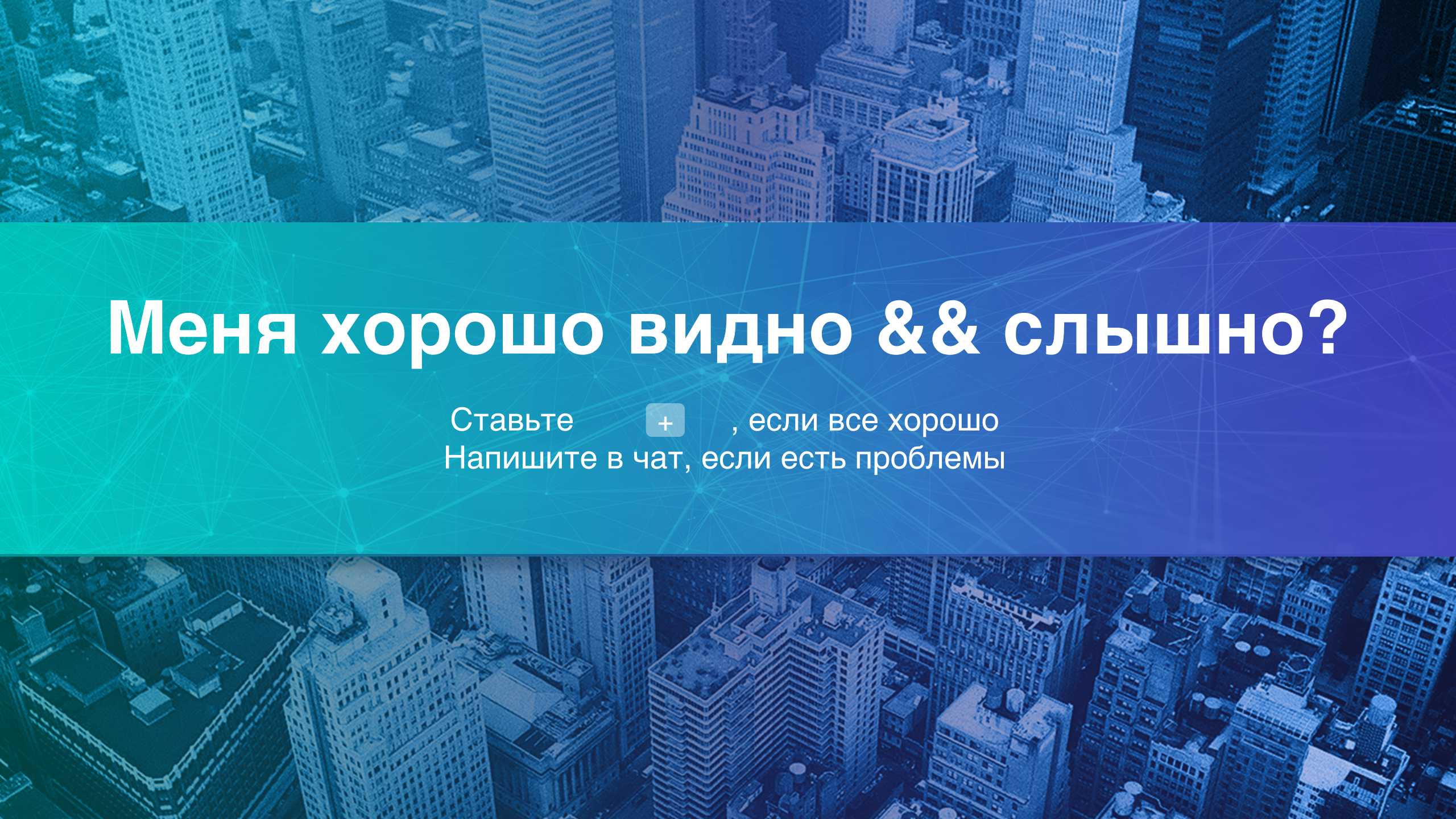
Проверить, идет ли запись!



Меня хорошо видно && слышно?

Ставьте + , если все хорошо
Напишите в чат, если есть проблемы

Цели вебинара

1. Разобраться что такое Keys. Какие они бывают. И для чего они нужны?

2. Рассмотреть различные Builders и как их применять.

3. Разобрать ДЗ.

Keys

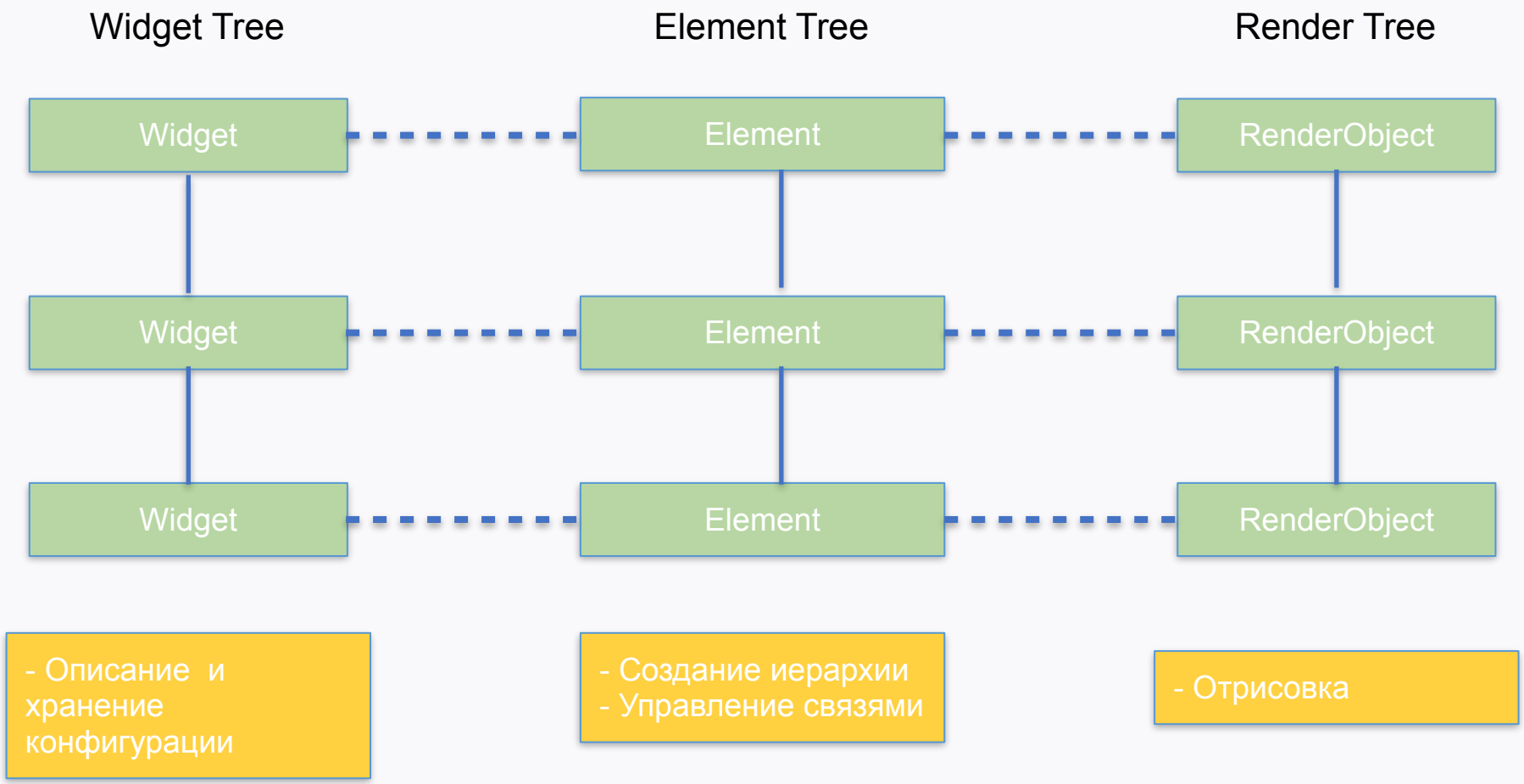
Keys

Key - это идентификатор для Widget и Element.

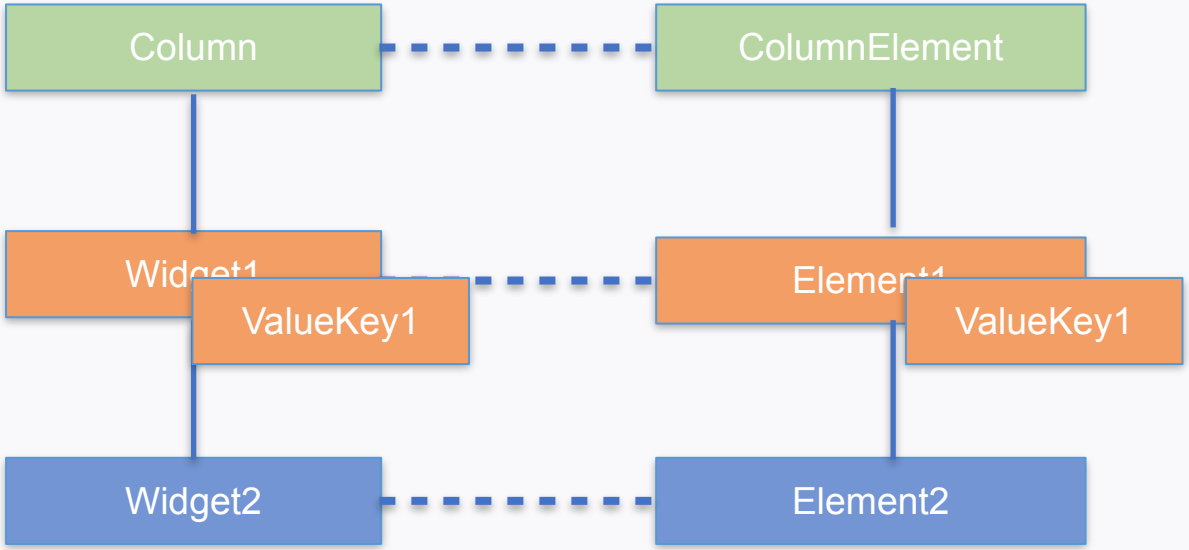
Новый **виджет** будет использоваться для обновления существующего **элемента**,

только если его ключ то же самое, что и ключ текущего виджета, связанного с элементом.

Keys

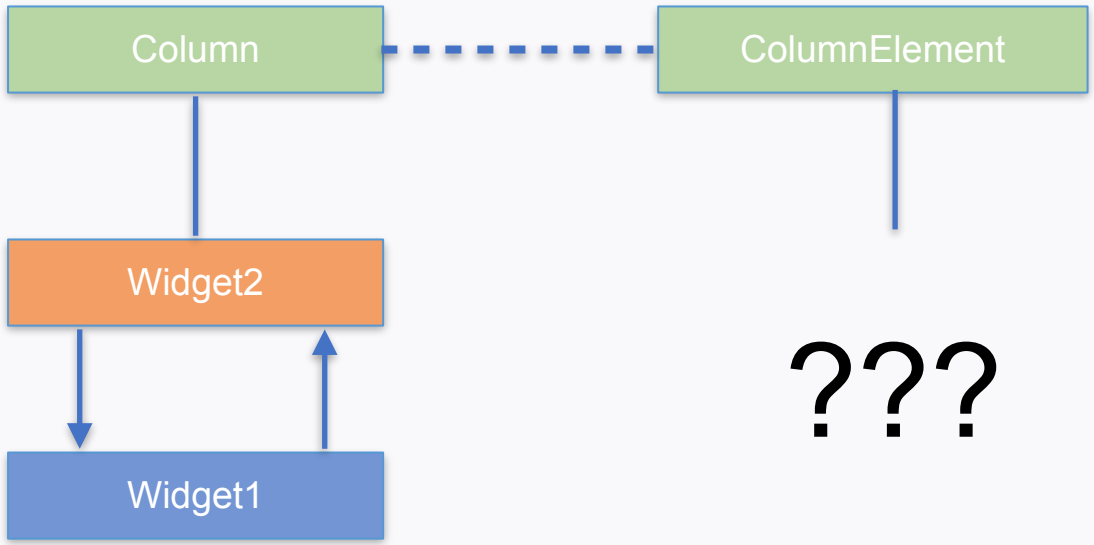

Widget Tree
Element Tree
Render Tree
Widget
Widget
Widget
Element
Element
Element
RenderObject
RenderObject
RenderObject
- Описание и хранение конфигурации
- Создание иерархии
- Управление связями
- Отрисовка

Keys


Column
ColumnElement
Widget1
ValueKey1
Element1
ValueKey1
Widget2
Element2

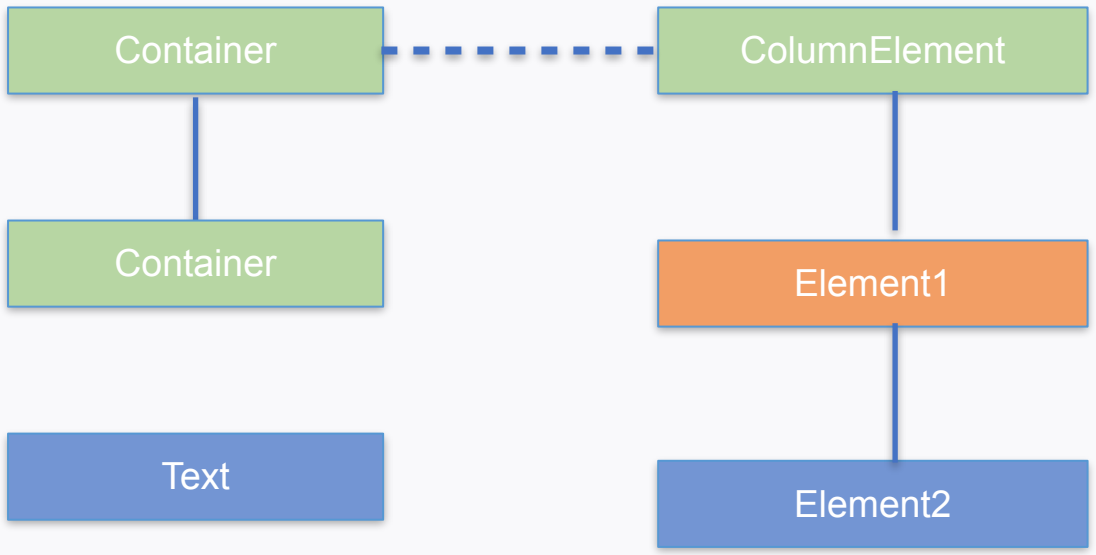
Keys

Что будет есть Text1 и Text2 поменять местами?




Смотрим код

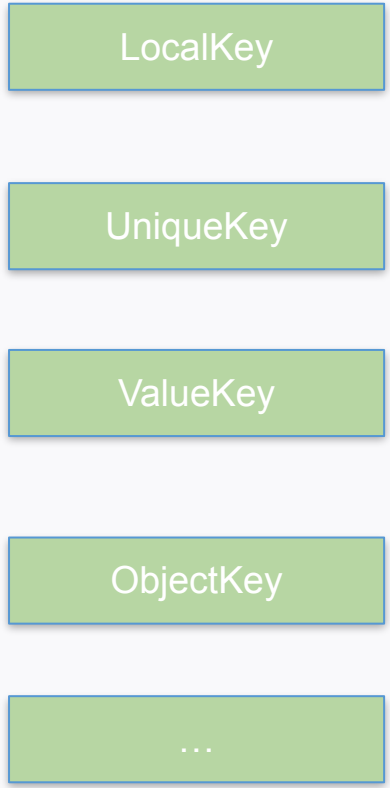
Keys


Container
ColumnElement
Container
Element1
Text
Element2

LocalKeys

LocalKey - ключ который нужен **только** для сопоставления Widget и Element.

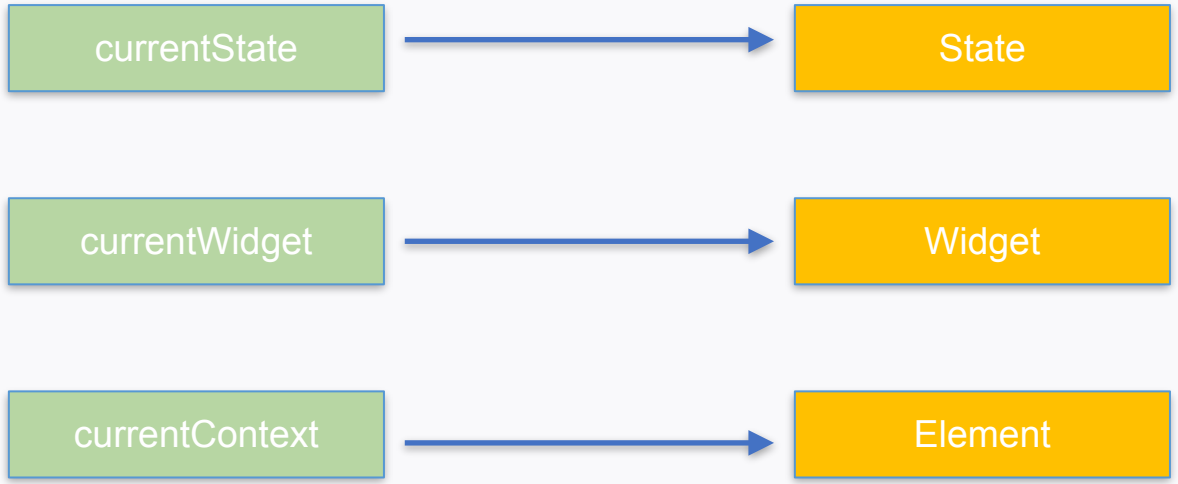
Не хранит в себе никакой информации



GlobalKey

Обеспечивает доступ к объектам связанных с Element.

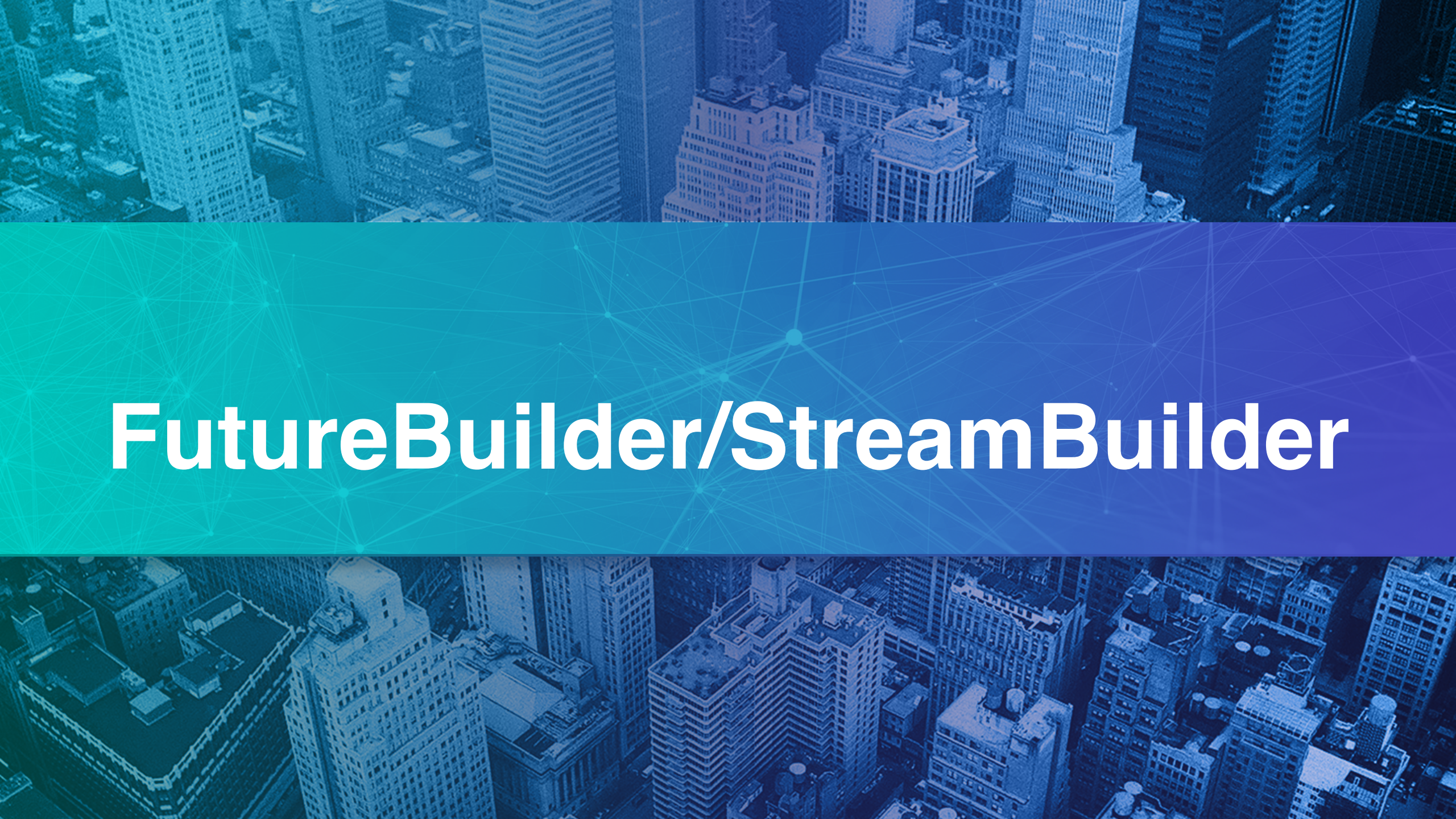
Через него можно получить



Builders

Builders

FutureBuilder	Обновляет состояние виджета в зависимости от результата Future
StreamBuilder	Обновляет состояние виджета в зависимости от последнего значения в Stream
Builder	Обертка которая позволяет получить дочерний BuildContext внутри текущего виджета
LayoutBuilder	Помогает понять constraints при построении виджета
ValueListenableBuilder	Обновляет состояние виджета в зависимости от подписки на
…	


FutureBuilder/StreamBuilder

FutureBuilder / StreamBuilder

Виджет, который строится на основе AsyncSnapshot

FutureBuilder<T>()

<T> - тип данных асинхронного вычисления

future: Future<T>

initialData: T()

builder:
- Widget Function(BuildContext context,AsyncSnapshot<T> snapshot)

StreamBuilder<T>()

<T> - тип данных асинхронного вычисления

stream: Stream<T>

initialData: T()

builder:
- Widget Function(BuildContext context,AsyncSnapshot<T> snapshot)

AsyncSnapshot

Неизменяемое представление последнего взаимодействия с асинхронным вычислением.

ConnectionState

none	В настоящее время не подключен к каким-либо асинхронным вычислениям
waiting	Подключено к асинхронным вычислениям и ожидает взаимодействия
active	Подключен к активному асинхронному вычислению.
done	Подключен к завершенному асинхронному вычислению.


LayoutBuilder

LayoutBuilder

Позволяет построить Widget опираясь на BoxConstraints полученные от родителя

LayoutBuilder

builder: Widget Function(BuildContext context,BoxConstraints constraints)

BoxConstraints - ограничения которые задаются 4-мя параметрами.

minWidth

maxWidth

minHeight

maxHeight


ValueListenableBuilder

ValueListenableBuilder

Строит виджет исходя из текущего значения полученного от ValueListenable<T>

ValueListenableBuilder<T>

valueListenable: ValueListenable<T>

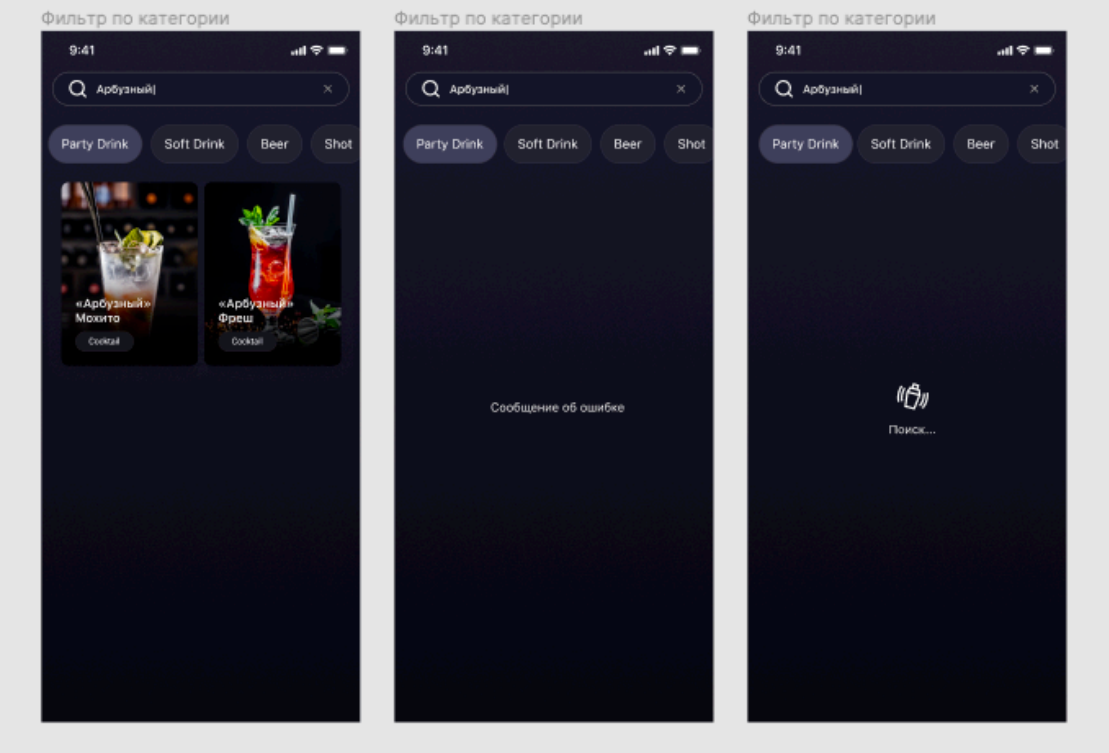
child: Widget

builder: Widget Function(BuildContext context, T value, Widget child)

Child: - Используется если у нас есть Widget который не зависит от ValueListenable, но нам нужно его встроить в дерево виджетов внутри builder. Это хорошая практика с точки зрения производительности.

ДЗ

ДЗ



Сделать экран Фильтр по категории

1. Поисковую строку можно пока что не делать.
2. Фильтры это CocktailCategory
3. Данные получаем по сети через AsyncCocktailRepository().fetchCocktailsByCocktailCategory()
4.Используем StreamBuilder/FutureBuilder
5. Для скролла используем CustomScrollView
6. Остальные детали есть в Readme

Делаем fork от репозитория и сдаем через PR

Задание лежит тут lesson_07/homework/lib/ui/filter_page.dart

Ссылка на макет

Ссылка на репозиторий

Цели вебинара

1. Разобраться что такое Keys. Какие они бывают. И для чего они нужны?

2. Рассмотреть различные Builders и как их применять.

3. Разобрать ДЗ.