#13 | Lanjutan State Management dengan Streams

| **Summary** | Mampu mengelola state dengan streams dan BLoC pattern di Flutter |
| --- | --- |
| **URL** | 13-state-streams-bloc |
| **Category** | Mobile Flutter |
| **Environment** | web |
| **Status** | Published |
| **Feedback Link** | https://github.com/jti-polinema/flutter-codelab/issues |
| **Author** | Habibie Ed Dien |
| **Author LDAP** | hbb-polinema |
| **Analytics Account** | G-H7507PFEJB |

[Lanjutan State Management dengan Streams](#_ok7k5uux6)

[Tujuan Praktikum](#_21yzqg98x7h6)

[Sumber Daya yang Dibutuhkan](#_dusjnkkj26ob)

[Pengetahuan yang Anda harus Miliki](#_319h1e1sjzdz)

[Pendahuluan Dart Streams](#_t0tqoph2ohlh)

[Praktikum 1: Dart Streams](#_vmpz7ap0erhd)

[Langkah 1: Buat Project Baru](#_spr93e7x3yo2)

[Langkah 2: Buka file main.dart](#_2q85gi9dhaiq)

[Soal 1](#_bb1snip8ruyk)

[Langkah 3: Buat file baru stream.dart](#_p0t5r1yzb5qp)

[Langkah 4: Tambah variabel colors](#_xmhut8m5gdv1)

[Soal 2](#_ttbzdbbh92bd)

[Langkah 5: Tambah method getColors()](#_wtc8ac276zx5)

[Langkah 6: Tambah perintah yield\*](#_rkl6d7ct1s8j)

[Soal 3](#_i07c10rrlh5d)

[Langkah 7: Buka main.dart](#_sf55rqaqaz6r)

[Langkah 8: Tambah variabel](#_wq5i5ww5j4xa)

[Langkah 9: Tambah method changeColor()](#_rhi6kwkqddsm)

[Langkah 10: Lakukan override initState()](#_kgve4w76y3z3)

[Langkah 11: Ubah isi Scaffold()](#_we6tjrvb0pu2)

[Langkah 12: Run](#_wpuhk0pub24v)

[Soal 4](#_rarblsa55tt1)

[Langkah 13: Ganti isi method changeColor()](#_q9klfvxqpzx2)

[Soal 5](#_xc5ntuz7143u)

[Praktikum 2: Stream controllers dan sinks](#_1e543m1dm1c4)

[Langkah 1: Buka file stream.dart](#_u3l46a7g73v0)

[Langkah 2: Tambah class NumberStream](#_gfa7d3gyzo4o)

[Langkah 3: Tambah StreamController](#_wc7bdoy0rvka)

[Langkah 4: Tambah method addNumberToSink](#_dxl4engxhz7o)

[Langkah 5: Tambah method close()](#_oi2maeynzj9o)

[Langkah 6: Buka main.dart](#_mdpi9i7qep2r)

[Langkah 7: Tambah variabel](#_finb1ju2tytk)

[Langkah 8: Edit initState()](#_n1owjg1xgot9)

[Langkah 9: Edit dispose()](#_wgcvtobmc11h)

[Langkah 10: Tambah method addRandomNumber()](#_9ti7kzsslv4y)

[Langkah 11: Edit method build()](#_cmbkp9a8xgul)

[Langkah 12: Run](#_99ufg6c4owg)

[Soal 6](#_kbavju8eav8k)

[Langkah 13: Buka stream.dart](#_z6165rumyro0)

[Langkah 14: Buka main.dart](#_wl9fnmh670ox)

[Langkah 15: Edit method addRandomNumber()](#_d1a83arnbkuj)

[Soal 7](#_nmtnybz24m4n)

[Praktikum 3: Injeksi data ke streams](#_riuqbijaf0zt)

[Langkah 1: Buka main.dart](#_uv7foq4euher)

[Langkah 2: Tambahkan kode ini di initState](#_di3ldp29lo3b)

[Langkah 3: Tetap di initState](#_ma6gko34okso)

[Langkah 4: Run](#_oxnid84td09d)

[Soal 8](#_icte5u7gn468)

[Praktikum 4: Subscribe ke stream events](#_1c5hdgpeva1a)

[Langkah 1: Tambah variabel](#_mf0b60ekdgn2)

[Langkah 2: Edit initState()](#_hyt5pupp74tf)

[Langkah 3: Tetap di initState()](#_45fx8aigkiyt)

[Langkah 4: Tambah properti onDone()](#_azsb3vk7tfot)

[Langkah 5: Tambah method baru](#_u8zmxi9ax5h1)

[Langkah 6: Pindah ke method dispose()](#_1uddj4fwvvh9)

[Langkah 7: Pindah ke method build()](#_bbnibecwo5bx)

[Langkah 8: Edit method addRandomNumber()](#_1eglzmupw5yp)

[Langkah 9: Run](#_r0j41tes71rv)

[Langkah 10: Tekan button ‘Stop Subscription’](#_12vta3m65u3v)

[Soal 9](#_no5m83l02chs)

[Praktikum 5: Multiple stream subscriptions](#_3fcxszkylq4h)

[Langkah 1: Buka file main.dart](#_o8guex945zrp)

[Langkah 2: Edit initState()](#_6z67xj3j5wqh)

[Langkah 3: Run](#_hu872j19rioc)

[Soal 10](#_xp1s9rcn8274)

[Langkah 4: Set broadcast stream](#_2fs1ku91ieim)

[Langkah 5: Edit method build()](#_1ajdyynpvb6b)

[Langkah 6: Run](#_7it4mvvlwwm)

[Soal 11](#_vgspoagmptbp)

[Praktikum 6: StreamBuilder](#_yocqwzbhogdc)

[Langkah 1: Buat Project Baru](#_fhnygqckvfg)

[Langkah 2: Buat file baru stream.dart](#_xgvweet1rjak)

[Langkah 3: Tetap di file stream.dart](#_ai3lmohajpie)

[Langkah 4: Edit main.dart](#_gcxu61he253x)

[Langkah 5: Tambah variabel](#_pqxdjfvtezod)

[Langkah 6: Edit initState()](#_8fpvkqkqcuy2)

[Langkah 7: Edit method build()](#_1xpuqms9u3kz)

[Langkah 8: Run](#_d4s7t7kac8ul)

[Soal 12](#_w2bg8hxbwg58)

[Praktikum 7: BLoC Pattern](#_z1vcfco4xqur)

[Langkah 1: Buat Project baru](#_j8ii7uwgw1qx)

[Langkah 2: Isi kode random\_bloc.dart](#_jpb7v8sxomdn)

[Langkah 3: Buat class RandomNumberBloc()](#_d2p9iyg8m9n5)

[Langkah 4: Buat variabel StreamController](#_ni8oooih0sc8)

[Langkah 5: Buat constructor](#_lmjgavevz3uj)

[Langkah 6: Buat method dispose()](#_axbkqnc24e2v)

[Langkah 7: Edit main.dart](#_frwx0mqs2mlt)

[Langkah 8: Buat file baru random\_screen.dart](#_lqtycge6h8jw)

[Langkah 9: Lakukan impor material dan random\_bloc.dart](#_ar5f5kofgkjk)

[Langkah 10: Buat StatefulWidget RandomScreen](#_qmbx8t4zia5p)

[Langkah 11: Buat variabel](#_xzhuzrpiphmu)

[Langkah 12: Buat method dispose()](#_4khqqko84jxt)

[Langkah 13: Edit method build()](#_azcexannwvbl)

[Soal 13](#_3z4fyksrp32d)

[Selamat!](#_s22qmcj4zu61)

[Apa selanjutnya?](#_9mqifffit2ew)

[Referensi](#_an95hoqxrlhg)

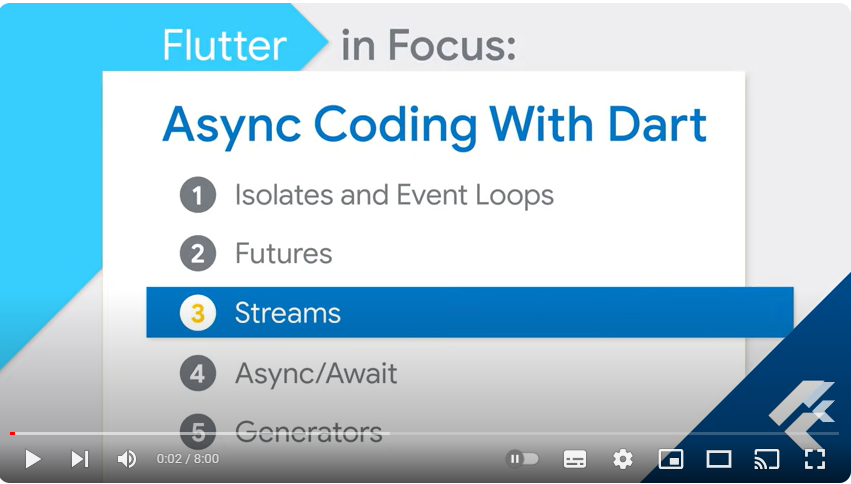
# Lanjutan State Management dengan Streams

**Terakhir diperbarui:** 20 November 2023

**Penulis:** Habibie Ed Dien

Pada codelab ini, Anda akan mempelajari tentang streams dan BLoC di Flutter beserta contoh penggunaannya. Cara kerja, manfaat, dan cara mengelola state lebih lanjut.

Video berikut menjelaskan tentang apa itu Streams dan BLoC serta bagaimana manfaatnya terhadap pengelolaan state. Silakan simak dan pahami!



## Tujuan Praktikum

Setelah menyelesaikan codelab ini Anda akan mampu untuk:

* Menggunakan streams
* Menggunakan stream controllers dan sinks
* Menginjeksi transformasi data ke dalam streams
* Mendaftar di stream events
* Menggunakan multi stream subscriptions
* Menggunakan StreamBuilder untuk membuat UI reactive
* Menggunakan BLoC pattern

| **Catatan:** Materi ini diadaptasi dari Buku “Flutter Cookbook, 100+ step-by-step recipes for building cross-platform, professional grade apps with Flutter 3.10.x and Dart 3.x - Second Edition” oleh Simone Alessandria (2023). |
| --- |

## Sumber Daya yang Dibutuhkan

Berikut merupakan sumber daya yang diperlukan untuk menyelesaikan praktikum ini:

* PC atau Laptop dengan spesifikasi minimum RAM 4GB Processor Core i3
* Koneksi internet minimal kecepatan 1 Mbps
* Chrome Browser
* Perangkat mobile (Android/iOS)

## Pengetahuan yang Anda harus Miliki

* Algoritma dan Pemrograman Dasar
* Pemrograman Berorientasi Objek
* Bahasa Pemrograman Dart Dasar
* Teknik Dasar Pemrograman Asynchronous

# Pendahuluan Dart Streams

Duration: 10:00

Di Dart dan Flutter, future dan stream merupakan tool utama untuk melakukan pemrograman asynchronous. Future merepresentasikan single value yang diterima di waktu yang akan datang, sedangkan Stream adalah sebuah set (*sequence*) yang bisa memiliki nilai 0 atau lainnya dikirim secara asinkronus secara terus-menerus. Intinya, Stream merupakan aliran data yang dikirim terus-menerus.

Untuk mendapatkan data dari Stream, Anda harus melakukan subscribe (listen). Setiap kali data itu diterima, Anda mendapatkan data tersebut dan dapat dimanipulasi sesuai kebutuhan aplikasi.

| **Catatan:** Secara default, setiap stream hanya mengizinkan *single subscription*. Namun, Anda dapat juga mengaktifkan multiple subscription, nanti Anda akan lakukan pada praktikum 5. |
| --- |

## 

Pada codelab ini, Anda akan fokus bekerja dengan berbagai studi kasus pada stream di Flutter. Anda akan menggunakan stream dengan berbagai skenario berbeda, cara menulis dan membaca data ke stream, membangun UI berdasarkan stream, dan menggunakan pola BLoC untuk state management.

Seperti halnya Future, stream juga dapat menghasilkan data atau error, sehingga Anda akan melakukan praktikum seperti halnya Future di pekan lalu.

# Praktikum 1: Dart Streams

Duration: 20:00

Selesaikan langkah-langkah praktikum berikut ini menggunakan editor Visual Studio Code (VS Code) atau Android Studio atau code editor lain kesukaan Anda. Jawablah di laporan praktikum Anda (ketik di **README.md**) pada setiap soal yang ada di beberapa langkah praktikum ini.

| **Perhatian:** Diasumsikan Anda telah berhasil melakukan setup environment Flutter SDK, VS Code, Flutter Plugin, dan Android SDK pada pertemuan pertama. |
| --- |

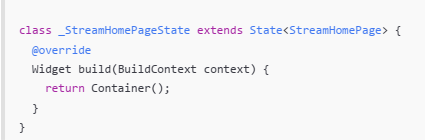
## Langkah 1: Buat Project Baru

Buatlah sebuah project flutter baru dengan nama **stream\_nama** (beri nama panggilan Anda) di folder **week-13/src/** repository GitHub Anda.

## Langkah 2: Buka file main.dart

Ketiklah kode seperti berikut ini.





| Soal 1  * Tambahkan **nama panggilan Anda** pada title app sebagai identitas hasil pekerjaan Anda. * Gantilah warna tema aplikasi sesuai kesukaan Anda. * Lakukan commit hasil jawaban Soal 1 dengan pesan **“W13: Jawaban Soal 1”** |
| --- |

## Langkah 3: Buat file baru stream.dart

Buat file baru di folder lib project Anda. Lalu isi dengan kode berikut.



## Langkah 4: Tambah variabel colors

Tambahkan variabel di dalam class ColorStream seperti berikut.



| Soal 2  * Tambahkan 5 warna lainnya sesuai keinginan Anda pada variabel colors tersebut. * Lakukan commit hasil jawaban Soal 2 dengan pesan **“W13: Jawaban Soal 2”** |
| --- |

## Langkah 5: Tambah method getColors()

Di dalam class ColorStream ketik method seperti kode berikut. Perhatikan tanda bintang di akhir keyword async\* (ini digunakan untuk melakukan Stream data)



## Langkah 6: Tambah perintah yield\*

Tambahkan kode berikut ini.

| yield\* Stream.periodic(  const Duration(seconds: 1), (int t) {  int index = t % colors.length;  return colors[index];  }); |
| --- |

| Soal 3  * Jelaskan fungsi keyword yield\* pada kode tersebut! * Apa maksud isi perintah kode tersebut? * Lakukan commit hasil jawaban Soal 3 dengan pesan **“W13: Jawaban Soal 3”** |
| --- |

## Langkah 7: Buka main.dart

Ketik kode impor file ini pada file main.dart

| import 'stream.dart'; |
| --- |

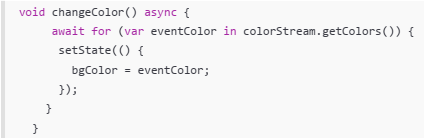
## Langkah 8: Tambah variabel

Ketik dua properti ini di dalam class \_StreamHomePageState



## Langkah 9: Tambah method changeColor()

Tetap di file main, Ketik kode seperti berikut



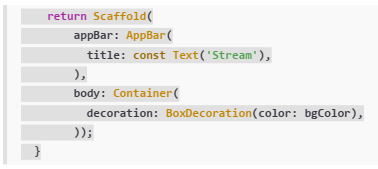
## **Langkah** 10: Lakukan override initState()

Ketika kode seperti berikut



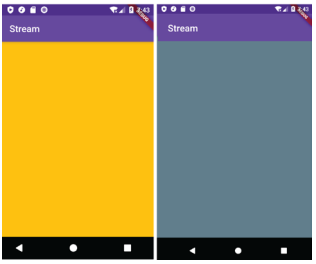
## **Langkah** 11: Ubah isi Scaffold()

Sesuaikan kode seperti berikut.



## Langkah 12: Run

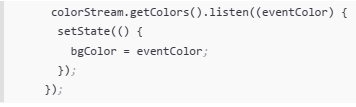
Lakukan running pada aplikasi Flutter Anda, maka akan terlihat berubah warna background setiap detik.



| Soal 4  * Capture hasil praktikum Anda berupa GIF dan lampirkan di README. * Lakukan commit hasil jawaban Soal 4 dengan pesan **“W13: Jawaban Soal 4”** |
| --- |

## Langkah 13: Ganti isi method changeColor()

Anda boleh comment atau hapus kode sebelumnya, lalu ketika kode seperti berikut.



| Soal 5  * Jelaskan perbedaan menggunakan listen dan await for (langkah 9) ! * Lakukan commit hasil jawaban Soal 5 dengan pesan **“W13: Jawaban Soal 5”** |
| --- |

| **Catatan:** Stream di Flutter memiliki fitur yang powerfull untuk menangani data secara async. Stream dapat dimanfaatkan pada skenario dunia nyata seperti real-time messaging, unggah dan unduh file, tracking lokasi user, bekerja dengan data sensor IoT, dan lain sebagainya. |
| --- |

## 

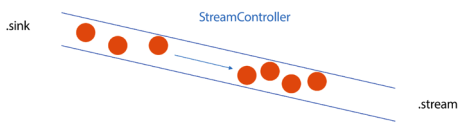
# Praktikum 2: Stream controllers dan sinks

Duration: 15:00

StreamControllers akan membuat jembatan antara Stream dan Sink. Stream berisi data secara sekuensial yang dapat diterima oleh subscriber manapun, sedangkan Sink digunakan untuk mengisi (injeksi) data.

Secara sederhana, StreamControllers merupakan stream management. Ia akan otomatis membuat stream dan sink serta beberapa method untuk melakukan kontrol terhadap event dan fitur-fitur yang ada di dalamnya.

Anda dapat membayangkan stream sebagai pipa air yang mengalir searah, dari salah satu ujung Anda dapat mengisi data dan dari ujung lain data itu keluar. Anda dapat melihat konsep stream pada gambar diagram berikut ini.



Di Flutter, Anda dapat menggunakan StreamController untuk mengontrol aliran data stream. Sebuah StreamController memiliki sebuah properti bernama sink yang berguna untuk insert data. Sedangkan properti stream berguna untuk menerima atau keluarnya data dari StreamController tersebut.

Setelah Anda menyelesaikan praktikum 1, Anda dapat melanjutkan praktikum 2 ini. Selesaikan langkah-langkah praktikum berikut ini menggunakan editor Visual Studio Code (VS Code) atau Android Studio atau code editor lain kesukaan Anda. Jawablah di laporan praktikum Anda pada setiap soal yang ada di beberapa langkah praktikum ini.

| **Perhatian:** Diasumsikan Anda telah berhasil menyelesaikan Praktikum 1. |
| --- |

Pada codelab ini, kita akan menambah kode dari aplikasi **stream** di praktikum sebelumnya.

## Langkah 1: Buka file stream.dart

Lakukan impor dengan mengetik kode ini.

| import 'dart:async'; |
| --- |

## Langkah 2: Tambah class NumberStream

Tetap di file stream.dart tambah class baru seperti berikut.



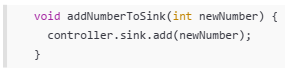
## Langkah 3: Tambah StreamController

Di dalam class NumberStream buatlah variabel seperti berikut.



## Langkah 4: Tambah method addNumberToSink

Tetap di class NumberStream buatlah method ini



## Langkah 5: Tambah method close()



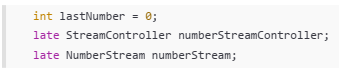
## **Langkah** 6: Buka main.dart

Ketik kode import seperti berikut

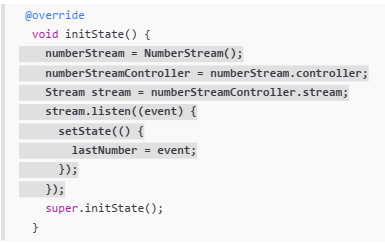


## **Langkah** 7: Tambah variabel

Di dalam class \_StreamHomePageState ketik variabel berikut



## Langkah 8: Edit initState()



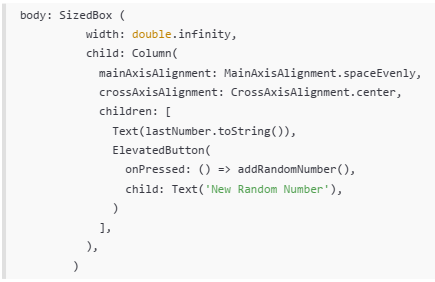
## Langkah 9: Edit dispose()



## Langkah 10: Tambah method addRandomNumber()

| void addRandomNumber() {  Random random = Random();  int myNum = random.nextInt(10);  numberStream.addNumberToSink(myNum);  } |
| --- |

## Langkah 11: Edit method build()



## Langkah 12: Run

Lakukan running pada aplikasi Flutter Anda, maka akan terlihat seperti gambar berikut.



| Soal 6  * Jelaskan maksud kode langkah 8 dan 10 tersebut! * Capture hasil praktikum Anda berupa GIF dan lampirkan di README. * Lalu lakukan commit dengan pesan “**W13: Jawaban Soal 6**”. |
| --- |

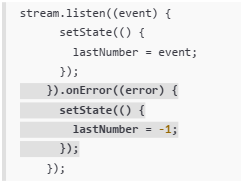
## Langkah 13: Buka stream.dart

Tambahkan method berikut ini.



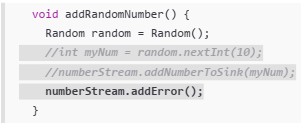
## Langkah 14: Buka main.dart

Tambahkan method onError di dalam class StreamHomePageState pada method listen di fungsi initState() seperti berikut ini.



## **Langkah** 15: Edit method addRandomNumber()

Lakukan *comment* pada dua baris kode berikut, lalu ketik kode seperti berikut ini.



| Soal 7  * Jelaskan maksud kode langkah 13 sampai 15 tersebut! * Kembalikan kode seperti semula pada Langkah 15, comment addError() agar Anda dapat melanjutkan ke praktikum 3 berikutnya. * Lalu lakukan commit dengan pesan “**W13: Jawaban Soal 7**”. |
| --- |

# Praktikum 3: Injeksi data ke streams

Duration: 15:00

Skenario yang umum dilakukan adalah melakukan manipulasi atau transformasi data stream sebelum sampai pada UI end user. Hal ini sangatlah berguna ketika Anda membutuhkan untuk filter data berdasarkan kondisi tertentu, melakukan validasi data, memodifikasinya, atau melakukan proses lain yang memicu beberapa output baru. Contohnya melakukan konversi angka ke string, membuat sebuah perhitungan, atau menghilangkan data yang berulang terus tampil.

Pada praktikum 3 ini, Anda akan menggunakan StreamTransformers ke dalam stream untuk melakukan map dan filter data.

Setelah Anda menyelesaikan praktikum 2, Anda dapat melanjutkan praktikum 3 ini. Selesaikan langkah-langkah praktikum berikut ini menggunakan editor Visual Studio Code (VS Code) atau Android Studio atau code editor lain kesukaan Anda. Jawablah di laporan praktikum Anda pada setiap soal yang ada di beberapa langkah praktikum ini.

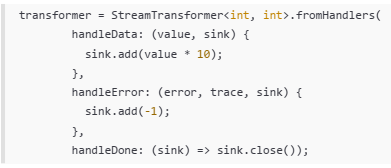
| **Perhatian:** Diasumsikan Anda telah berhasil menyelesaikan Praktikum 2. |
| --- |

## Langkah 1: Buka main.dart

Tambahkan variabel baru di dalam class \_StreamHomePageState



## Langkah 2: Tambahkan kode ini di initState



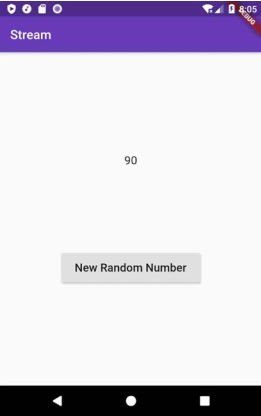
## Langkah 3: Tetap di initState

Lakukan edit seperti kode berikut.



## Langkah 4: Run

Terakhir, **run** atau tekan **F5** untuk melihat hasilnya jika memang belum running. Bisa juga lakukan **hot restart** jika aplikasi sudah running. Maka hasilnya akan seperti gambar berikut ini. Anda akan melihat tampilan angka dari 0 hingga 90.



| Soal 8  * Jelaskan maksud kode langkah 1-3 tersebut! * Capture hasil praktikum Anda berupa GIF dan lampirkan di README. * Lalu lakukan commit dengan pesan “**W13: Jawaban Soal 8**”. |
| --- |

# Praktikum 4: Subscribe ke stream events

Duration: 15:00

Dari praktikum sebelumnya, Anda telah menggunakan method listen mendapatkan nilai dari stream. Ini akan menghasilkan sebuah Subscription. Subscription berisi method yang dapat digunakan untuk melakukan listen pada suatu event dari stream secara terstruktur.

Pada praktikum 4 ini, kita akan gunakan Subscription untuk menangani event dan error dengan teknik praktik baik (*best practice*), dan menutup Subscription tersebut.

Setelah Anda menyelesaikan praktikum 3, Anda dapat melanjutkan praktikum 4 ini. Selesaikan langkah-langkah praktikum berikut ini menggunakan editor Visual Studio Code (VS Code) atau Android Studio atau code editor lain kesukaan Anda. Jawablah di laporan praktikum Anda pada setiap soal yang ada di beberapa langkah praktikum ini.

| **Perhatian:** Diasumsikan Anda telah berhasil menyelesaikan Praktikum 3. |
| --- |

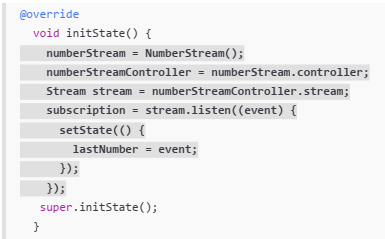
## Langkah 1: Tambah variabel

Tambahkan variabel berikut di class \_StreamHomePageState



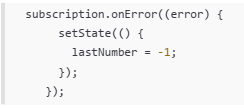
## Langkah 2: Edit initState()

Edit kode seperti berikut ini.



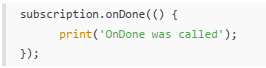
## Langkah 3: Tetap di initState()

Tambahkan kode berikut ini.



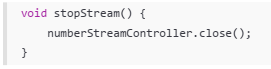
## Langkah 4: Tambah properti onDone()

Tambahkan dibawahnya kode ini setelah onError



## Langkah 5: Tambah method baru

Ketik method ini di dalam class \_StreamHomePageState



## Langkah 6: Pindah ke method dispose()

Jika method dispose() belum ada, Anda dapat mengetiknya dan dibuat override. Ketik kode ini didalamnya.



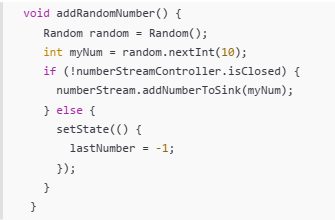
## Langkah 7: Pindah ke method build()

Tambahkan button kedua dengan isi kode seperti berikut ini.



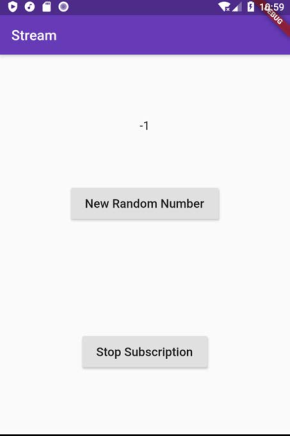
## Langkah 8: Edit method addRandomNumber()

Edit kode seperti berikut ini.



## Langkah 9: Run

Anda akan melihat dua button seperti gambar berikut.



## Langkah 10: Tekan button ‘Stop Subscription’

Anda akan melihat pesan di Debug Console seperti berikut.



| Soal 9  * Jelaskan maksud kode langkah 2, 6 dan 8 tersebut! * Capture hasil praktikum Anda berupa GIF dan lampirkan di README. * Lalu lakukan commit dengan pesan “**W13: Jawaban Soal 9**”. |
| --- |

# Praktikum 5: Multiple stream subscriptions

Duration: 13:00

Secara default, stream hanya bisa digunakan untuk satu subscription. Jika Anda mencoba untuk melakukan subscription yang sama lebih dari satu, maka akan terjadi error. Untuk menangani hal itu, tersedia **broadcast** stream yang dapat digunakan untuk multiple subscriptions. Pada praktikum ini, Anda akan mencoba untuk melakukan multiple stream subscriptions.

Setelah Anda menyelesaikan praktikum 4, Anda dapat melanjutkan praktikum 5 ini. Selesaikan langkah-langkah praktikum berikut ini menggunakan editor Visual Studio Code (VS Code) atau Android Studio atau code editor lain kesukaan Anda. Jawablah di laporan praktikum Anda pada setiap soal yang ada di beberapa langkah praktikum ini.

| **Perhatian:** Diasumsikan Anda telah berhasil menyelesaikan Praktikum 4. |
| --- |

## Langkah 1: Buka file main.dart

Ketik variabel berikut di class \_StreamHomePageState



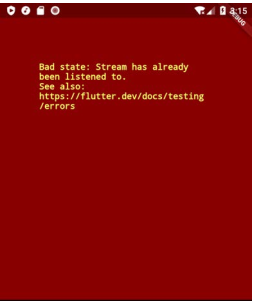
## Langkah 2: Edit initState()

Ketik kode seperti berikut.



## Langkah 3: Run

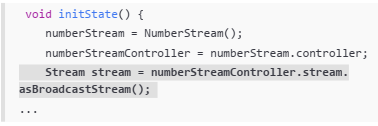
Lakukan run maka akan tampil error seperti gambar berikut.



| Soal 10  * Jelaskan mengapa error itu bisa terjadi ? |
| --- |

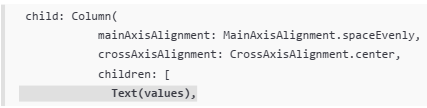
## Langkah 4: Set broadcast stream

Ketik kode seperti berikut di method initState()



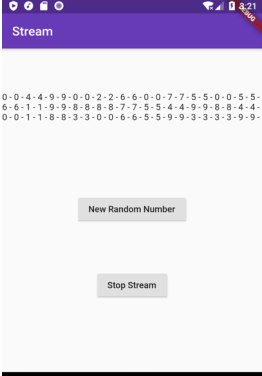
## Langkah 5: Edit method build()

Tambahkan text seperti berikut



## Langkah 6: Run

Tekan button ‘**New Random Number**’ beberapa kali, maka akan tampil teks angka terus bertambah sebanyak dua kali.



| Soal 11  * Jelaskan mengapa hal itu bisa terjadi ? * Capture hasil praktikum Anda berupa GIF dan lampirkan di README. * Lalu lakukan commit dengan pesan “**W13: Jawaban Soal 10,11**”. |
| --- |

# Praktikum 6: StreamBuilder

Duration: 15:00

StreamBuilder adalah sebuah widget untuk melakukan listen terhadap event dari stream. Ketika sebuah event terkirim, maka akan dibangun ulang semua turunannya. Seperti halnya widget FutureBuilder pada pertemuan pekan lalu, StreamBuilder berguna untuk membangun UI secara reaktif yang diperbarui setiap data baru tersedia.

Setelah Anda menyelesaikan praktikum 5, Anda dapat melanjutkan praktikum 6 ini. Selesaikan langkah-langkah praktikum berikut ini menggunakan editor Visual Studio Code (VS Code) atau Android Studio atau code editor lain kesukaan Anda. Jawablah di laporan praktikum Anda pada setiap soal yang ada di beberapa langkah praktikum ini.

| **Perhatian:** Diasumsikan Anda telah berhasil menyelesaikan Praktikum 5. |
| --- |

## Langkah 1: Buat Project Baru

Buatlah sebuah project flutter baru dengan nama **streambuilder\_nama** (beri nama panggilan Anda) di folder **week-13/src/** repository GitHub Anda.

## Langkah 2: Buat file baru stream.dart

Ketik kode ini



## Langkah 3: Tetap di file stream.dart

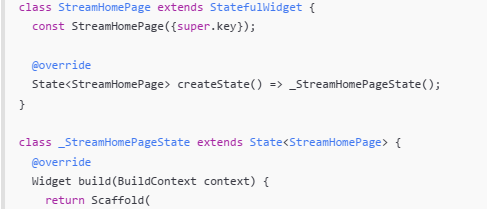
Ketik kode seperti berikut.

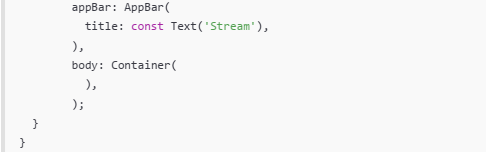


## Langkah 4: Edit main.dart

Ketik kode seperti berikut ini.







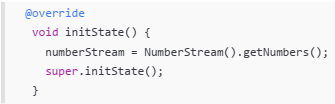
## Langkah 5: Tambah variabel

Di dalam class \_StreamHomePageState, ketika variabel ini.

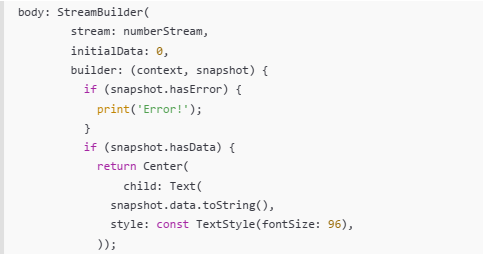


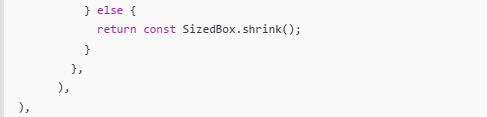
## Langkah 6: Edit initState()

Ketik kode seperti berikut.



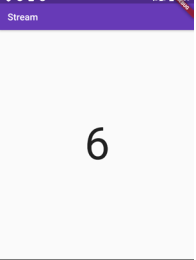
## Langkah 7: Edit method build()





## Langkah 8: Run

Hasilnya, setiap detik akan tampil angka baru seperti berikut.



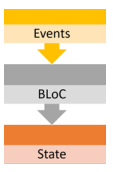
| Soal 12  * Jelaskan maksud kode pada langkah 3 dan 7 ! * Capture hasil praktikum Anda berupa GIF dan lampirkan di README. * Lalu lakukan commit dengan pesan “**W13: Jawaban Soal 12**”. |
| --- |

# Praktikum 7: BLoC Pattern

Duration: 15:00

Ketika menggunakan pola BLoC, maka segalanya merupakan stream event. BLoC atau **Business Logic Component** adalah lapisan antara semua sumber data dan UI yang mengonsumsi data itu. Contohnya seperti sumber data dari HTTP layanan web servis atau JSON dari sebuah basis data.

Sebuah BLoC menerima stream data dari sumbernya, proses itu membutuhkan logika bisnis Anda, dan return stream data ke *subscriber*-nya. Perhatikan diagram pola kerja BLoC berikut ini.



Alasan utama menggunakan BLoC adalah memisahkan logika bisnis aplikasi dengan presentasi UI pada widget, terutama akan sangat berguna ketika aplikasi Anda mulai semakin kompleks dan membutuhkan akses state di berbagai tempat. Hal ini akan membuat semakin mudah dalam menggunakan kode Anda, pada beberapa bagian di aplikasi atau bahkan berbeda aplikasi. Selain itu, BLoC secara independen berdiri sendiri dengan UI, sehingga sangat mudah dilakukan isolasi dalam proses testing.

Pada praktikum codelab ini, Anda akan membuat aplikasi sederhana menggunakan BLoC, namun Anda dapat dengan mudah mengembangkannya untuk aplikasi yang lebih besar ruang lingkupnya.

Setelah Anda menyelesaikan praktikum 6, Anda dapat melanjutkan praktikum 7 ini. Selesaikan langkah-langkah praktikum berikut ini menggunakan editor Visual Studio Code (VS Code) atau Android Studio atau code editor lain kesukaan Anda. Jawablah di laporan praktikum Anda pada setiap soal yang ada di beberapa langkah praktikum ini.

| **Perhatian:** Diasumsikan Anda telah berhasil menyelesaikan Praktikum 6. |
| --- |

## Langkah 1: Buat Project baru

Buatlah sebuah project flutter baru dengan nama **bloc\_random\_nama** (beri nama panggilan Anda) di folder **week-13/src/** repository GitHub Anda. Lalu buat file baru di folder lib dengan nama random\_bloc.dart

## Langkah 2: Isi kode random\_bloc.dart

Ketik kode impor berikut ini.

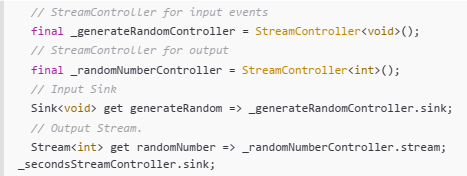


## Langkah 3: Buat class RandomNumberBloc()



## Langkah 4: Buat variabel StreamController

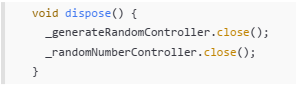
Di dalam class RandomNumberBloc() ketik variabel berikut ini



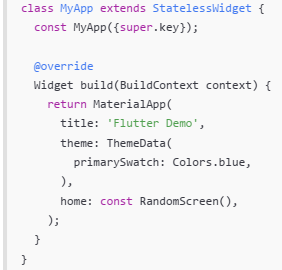
## Langkah 5: Buat constructor



## Langkah 6: Buat method dispose()



## Langkah 7: Edit main.dart



## Langkah 8: Buat file baru random\_screen.dart

Di dalam folder lib project Anda, buatlah file baru ini.

## Langkah 9: Lakukan impor material dan random\_bloc.dart

Ketik kode ini di file baru random\_screen.dart



## Langkah 10: Buat StatefulWidget RandomScreen

Buatlah di dalam file random\_screen.dart

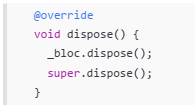
## Langkah 11: Buat variabel

Ketik kode ini di dalam class \_RandomScreenState



## Langkah 12: Buat method dispose()

Ketik kode ini di dalam class \_StreamHomePageState



## Langkah 13: Edit method build()

Ketik kode ini di dalam class \_StreamHomePageState



Run aplikasi, maka Anda akan melihat angka acak antara angka 0 sampai 9 setiap kali menekan tombol FloactingActionButton.

| Soal 13  * Jelaskan maksud praktikum ini ! Dimanakah letak konsep pola BLoC-nya ? * Capture hasil praktikum Anda berupa GIF dan lampirkan di README. * Lalu lakukan commit dengan pesan “**W13: Jawaban Soal 13**”. |
| --- |

# Selamat!

Duration: 0:00

Selamat Anda telah menyelesaikan Codelab ini. Anda telah mempelajari terkait stream, BLoC dan contoh penggunaannya.

Pada codelab berikutnya, Anda akan mempelajari tentang Persistensi Data.

Jangan sungkan jika Anda menemukan kesalahan pada codelab ini untuk merevisi atau sekedar melaporkan issue melalui tautan di pojok kiri bawah ([Report a mistake](https://github.com/jti-polinema/flutter-codelab/issues)).

## Apa selanjutnya?

Silakan cek beberapa sumber belajar lainnya…

* [Best practice guidance for live streaming programming](https://www.youtube.com/watch?v=mGkuOQK3kAM)
* <https://dart.dev/tutorials/language/streams>
* <https://dart.dev/articles/libraries/creating-streams>
* <https://api.dart.dev/stable/2.18.6/dart-async/Stream/map.html>
* <https://api.flutter.dev/flutter/dart-async/StreamController-class.html>
* <https://api.flutter.dev/flutter/widgets/StreamBuilder-class.html>
* <https://pub.dev/packages/flutter_bloc>
* <https://docs.flutter.dev/data-and-backend/state-mgmt/options>

## Referensi

* [Buku “Flutter Cookbook, 100+ step-by-step recipes for building cross-platform, professional grade apps with Flutter 3.10.x and Dart 3.x - Second Edition” oleh Simone Alessandria (2023)](https://drive.google.com/file/d/1W9uSLC9wlf9fFudmCWFHTx9dan5SXII0/view?usp=sharing)
* [Flutter for Beginners Second Edition](https://drive.google.com/file/d/1hJbctx-_O3B6kUeG70bJD5xfbnsWwBXY/view?usp=sharing)
* <https://www.youtube.com/flutterdev>
* <https://github.com/PacktPublishing/Flutter-for-Beginners-Second-Edition>