মাষ্টার অন্য ডার্ট ফাইল এবং এক্সেপশন হ্যান্ডলিং

এই ক্লাসে আমরা শিখবো Dart এ ফাইল নিয়ে কাজ করা এবং এরর হলে কীভাবে সুন্দরভাবে হ্যান্ডেল করা যায়। বাস্তব অ্যাপে ডেটা সংরক্ষণ, পড়া, ডিলিট করা বা কোনো সমস্যা হলে অ্যাপ ক্র্যাশ না করানোর জন্য এই টপিকগুলো খুব গুরুত্বপূর্ণ।

What we learn Today

01. Working with dart:io library

02. File read and write

03. File delete, rename, and existence check

04. Create and delete directories

05. List files inside a directory

06. Working with file paths

07. Exception vs Error (difference)

08. Using try-catch for error handling

09. on vs catch for specific exceptions

10. Using finally block

11. Throwing custom exceptions

12. Handling asynchronous errors

01. Working with dart:io library

dart:io কী?

dart:io হলো Dart-এর built-in লাইব্রেরি, যেটা দিয়ে আমরা:

- File নিয়ে কাজ করি
- Directory নিয়ে কাজ করি
- Input/Output handle করি

⚠️ Note:

dart:io Flutter web-এ কাজ করে না, mobile/desktop-এ কাজ করে।

Import করা


```
import 'dart:io';
```

Simple print


```
void main() {
  print(Directory.current.path);
}
```

02. File read and write

Basic idea

File-এ data:

- লিখতে পারি (write)
- পড়তে পারি (read)

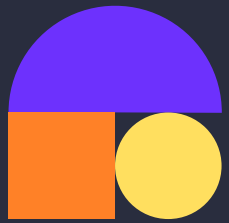
Write to file

File file = File('data.txt');
file.writeAsStringSync('Hello Flutter');

Read from file

File file = File('data.txt');
String content = file.readAsStringSync();
print(content);

03. File delete, rename, and existence check



Basic idea

File নিয়ে তিনটা **common** কাজ:

- আছে কিনা **check**
- **delete** করা
- **rename** করা

Check existence

```
File file =
File('data.txt');
print(file.existsSync());
```

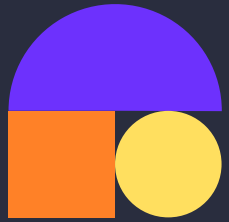
File file = File('data.txt');
print(file.existsSync());

Rename file

```
file.renameSync('new_data.txt')
;
```

file.renameSync('new_data.txt');

03.1. Add new string with existing file



Basic idea

- ডিফল্টভাবে writeAsString() → পুরোনো ডেটা মুছে নতুন ডেটা লেখে
-
- কিন্তু mode: FileMode.append ব্যবহার করলে →

  👉 পুরোনো ডেটার শেষে নতুন ডেটা যোগ হয়

Append new line

```
File file = File('data.txt');

await file.writeAsString(
  'This is new data\n',
  mode: FileMode.append,
);

print('Data appended successfully');
```

04. Create and delete directories

Basic idea

- Directory মানে folder

Check existence

```
Directory dir =
Directory('my_folder');
dir.createSync();
```

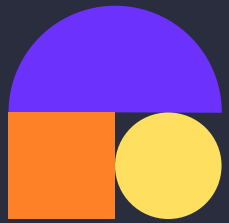
Directory dir = Directory('my_folder');
dir.createSync();

Rename file

```
dir.deleteSync()
;
```

dir.deleteSync();

05. List files inside a directory



Basic idea

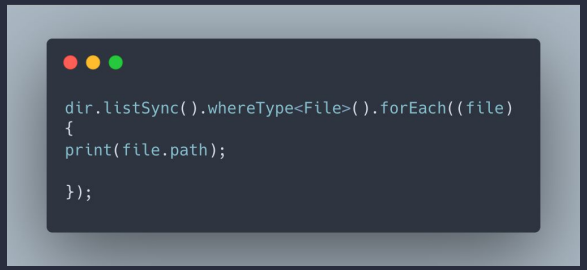
Folder এর ভিতরে কী কী **file** আছে, সেটা বের করা।

Check existence

```
Directory dir =
Directory('.');
dir.listSync().forEach((item)
{
  print(item.path);
});
```

```
Directory dir = Directory('.');
dir.listSync().forEach((item) {
  print(item.path);
});
```

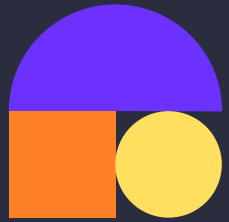
Only files



```
dir.listSync().whereType<File>().forEach((file)
{
print(file.path);

});
```

```
dir.listSync().whereType<File>().forEach((file) {

  print(file.path);

});
```

06. Working with file paths



Basic idea

Path মানে file বা folder এর address।

Get current path

```
print(Directory.current.path)
;
```

print(Directory.current.path);

Join path manually

```
File file =

File('${Directory.current.path}/data.txt')
;
```

File file =
File('${Directory.current.path}/data.txt');

06.1. Delete folder with all data

Basic idea

Directory নিজে নিজে খালি না হলে delete হয় না।
তাই আমাদের বলতে হয়:

> “এই directory-র ভিতরে যা আছে সব আগে মুছো, তারপর directory-টাও মুছো”

এটাই হলো recursive: true

Get current path

```
Directory parentDir = Directory('my_folder');

if (await parentDir.exists()) {
  await parentDir.delete(recursive: true);

  print('Parent directory and all contents deleted');
} else {
  print('Directory not found');

}
```


Directory parentDir = Directory('my_folder');

if (await parentDir.exists()) {
  await parentDir.delete(recursive: true);

  print('Parent directory and all contents deleted');

} else {
  print('Directory not found');

}

07. Exception vs Error (difference)

Basic idea

Exception	Error
Handle করা যায়	Usually handle করা যায় না
Runtime issue	Programming mistake
try-catch ব্যবহার হয়	App crash হতে পারে

Exception

```
int.parse('abc'); // FormatException
```

int.parse('abc'); // FormatException

Error

```
List list = [];
print(list[5]); // RangeError
```

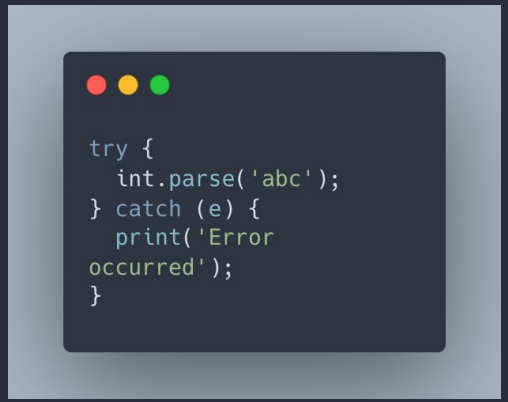
List list = [];
print(list[5]); // RangeError

08. Using try-catch for error handling

Basic idea

Error হলে **app crash** না করে **handle** করা।

Basic try-catch

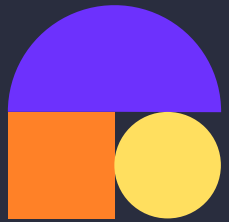



```
try {
  int.parse('abc');
} catch (e) {
  print('Error occurred');
}
```

File error handle


```
try {
  File('test.txt').readAsStringSync();
} catch (e) {
  print('File not found');
}
```

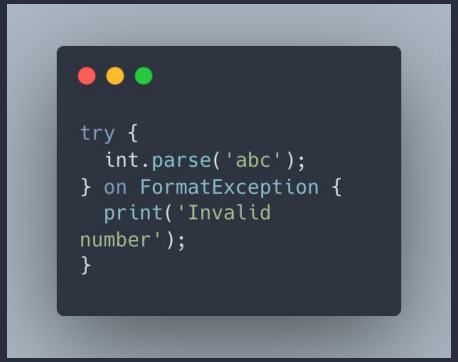
09. on vs catch for specific exceptions



Basic idea

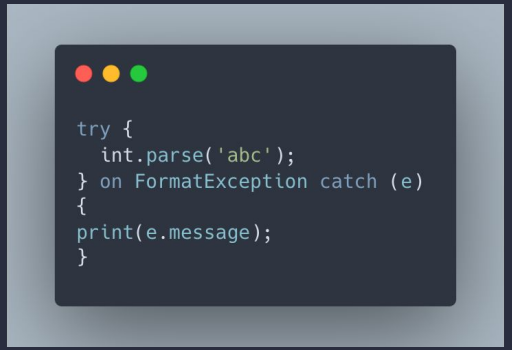
- **on** → নির্দিষ্ট **exception**
- **catch** → সব **exception**

on



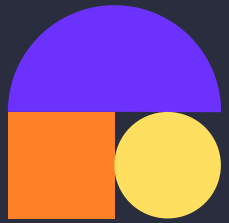

```
try {
  int.parse('abc');
} on FormatException {
  print('Invalid number');
}
```

on + catch




```
try {
  int.parse('abc');
} on FormatException catch (e) {
  print(e.message);
}
```

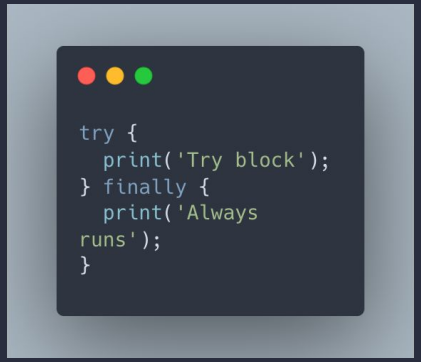
10. Using finally block



Basic idea

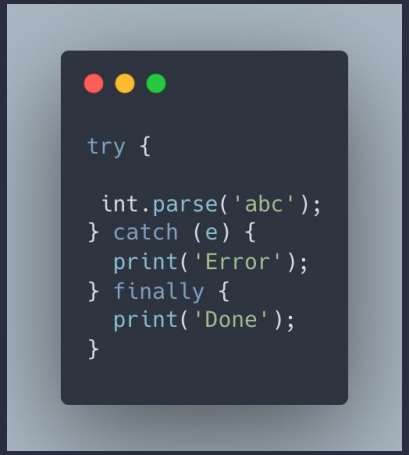
Error হোক বা না হোক, **finally** সব সময় execute হয়।

Example 1



```
try {
  print('Try block');
} finally {
  print('Always runs');
}
```

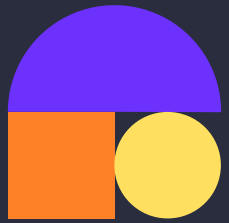
Example 2



```
try {

 int.parse('abc');
} catch (e) {
  print('Error');
} finally {
  print('Done');
}
```

```
try {
  int.parse('abc');
} catch (e) {
  print('Error');
} finally {
  print('Done');
}
```

11. Throwing custom exceptions



Basic idea

Future / async code-এ error handle করা।

async-await

```dart
try {
  await
File('test.txt').readAsString();
} catch (e) {
  print('Async error');
}
```

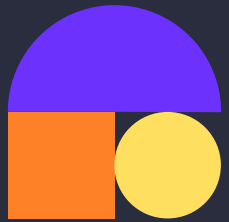
```dart
try {
  await File('test.txt').readAsString();
} catch (e) {
  print('Async error');
}
```

then-catchError

```dart
File('test.txt')
  .readAsString()
  .then((data) => print(data))
  .catchError((e) =>
print('Error'));
```

```dart
File('test.txt')
  .readAsString()
  .then((data) => print(data))
  .catchError((e) => print('Error'));
```

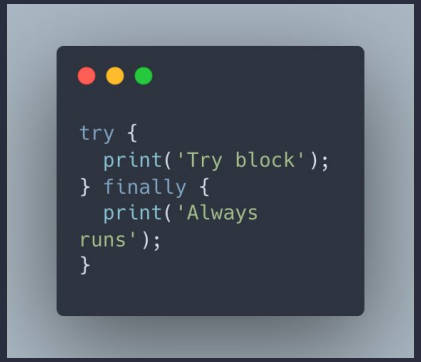
11. Throwing custom exceptions



Basic idea

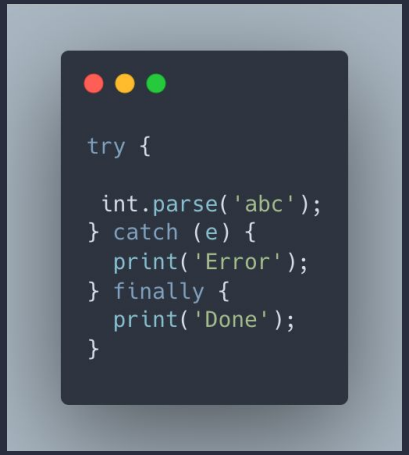
নিজের মতো করে **error** তৈরি করা।

Example 1




```
try {
  print('Try block');
} finally {
  print('Always runs');
}
```

Example 2




```
try {
  int.parse('abc');
} catch (e) {
  print('Error');
} finally {
  print('Done');
}
```

Assignment 1:
Basic File Write & Read

Objective: ফাইল তৈরি, লেখা এবং পড়া শেখা

Tasks:

1. notes.txt নামে একটি ফাইল তৈরি করো
2. ফাইলে নিজের নাম এবং বয়স লেখো
3. ফাইল থেকে ডেটা পড়ে console এ প্রিন্ট করো

Hint:

- getApplicationDocumentsDirectory()
- writeAsString()
- readAsString()

Assignment 2:
Check File Exists

Objective: ফাইল আছে কিনা চেক করা

Tasks:

1. data.txt নামে একটি ফাইল ধরো
2. ফাইলটি আছে কিনা চেক করো
3. - থাকলে: "File already exists" প্রিন্ট করো
   - না থাকলে: "File not found" প্রিন্ট করো

Hint:

- `file.exists()`

Assignment 3:
Append Data to File

Objective: আগের ডেটা না মুছে নতুন ডেটা যোগ করা

Tasks:

1. `log.txt` নামে একটি ফাইল নাও
2. প্রথমে `"App Started"` লেখো
3. আবার `"User Logged In"` append করো
4. পুরো ফাইলটি পড়ে প্রিন্ট করো

Hint:

- `FileMode.append`

Assignment 4:
File Delete with Exception Handling

Objective: try-catch ব্যবহার করা

Tasks:

1. `temp.txt` নামে একটি ফাইল ধরো
2. ফাইলটি delete করার চেষ্টা করো
3. - delete হলে `"File deleted successfully"`
   - error হলে `"Failed to delete file"`

Hint:

- `try-catch`
- `delete()`

Assignment 5:
Directory Operations

Objective: ডিরেক্টরি তৈরি ও চেক করা

Tasks:

1. my_folder নামে একটি directory তৈরি করো
2. directory আছে কিনা চেক করো
3. directory এর ভিতরে একটি ফাইল তৈরি করো

Hint:

- `Directory()`
- `create()`
- `exists()`

Advanced (Optional) Assignment:

Objective: Custom Exception বোঝা

Tasks:

1. InvalidAgeException নামে একটি custom exception তৈরি করো
2. ইউজার বয়স ১৮ এর নিচে হলে exception throw করো
3. main function এ সেটা catch করে message প্রিন্ট করো

Hint:

- `Directory()`
- `create()`
- `exists()`

আজ আমরা কি কি শিখলাম?

- ✔ ফাইল তৈরি, পড়া, লেখা
- ✔ ডিরেক্টরি ম্যানেজ করা
- ✔ **Runtime error** বুঝতে পারবে
- ✔ **try-catch** ব্যবহার করে অ্যাপ **crash** রোধ করবে
- ✔ **Real-world scenario** চিন্তা করতে পারবে

1. Dart File Handling কী?

File Handling মানে হলো:

- ফাইলে ডেটা লেখা
- ফাইল থেকে ডেটা পড়া
- ফাইল ডিলিট করা
- ফাইল আছে কিনা চেক করা

Dart এ এই কাজগুলো করার জন্য আমরা মূলত dart:io লাইব্রেরি ব্যবহার করি।

2. dart:io লাইব্রেরি

dart:io লাইব্রেরির মাধ্যমে আমরা পাই:

- File → ফাইল নিয়ে কাজ করতে
- Directory → ফোল্ডার নিয়ে কাজ করতে
- IOException → ফাইল সংক্রান্ত এরর ধরতে

এই লাইব্রেরি Flutter mobile app এর backend logic এবং server-side Dart এ বেশি ব্যবহৃত হয়।

✍️ 3. File Read & Write

আমরা পারি:

- ফাইলে লেখা → writeAsString()
- ফাইল থেকে পড়া → readAsString()

এগুলো সাধারণত:

- লগ সংরক্ষণ
- লোকাল ডেটা সেভ
- কনফিগারেশন ফাইল রাখতে ব্যবহৃত হয়

🗑️ 4. File Delete, Rename ও Exists Check

আমরা শিখবো:

- ফাইল ডিলিট করা → delete()
- ফাইলের নাম পরিবর্তন → rename()
- ফাইল আছে কিনা চেক → exists()

এগুলো বাস্তব অ্যাপে ব্যবহার হয়:

- অপ্রয়োজনীয় ডেটা পরিষ্কার করতে
- পুরোনো ফাইল ম্যানেজ করতে

📁 5. Directory Handling

Directory মানে ফোল্ডার।

আমরা শিখবো:

- ডিরেক্টরি তৈরি করা
- ডিরেক্টরি ডিলিট করা
- ডিরেক্টরির ভিতরে ফাইল লিস্ট বের করা

এটা কাজে লাগে:

- ফাইল গুছিয়ে রাখতে
- ইউজার ভিত্তিক ডেটা আলাদা করতে

6. File Path নিয়ে কাজ করা

ফাইল সবসময় একটি path এ থাকে।

আমরা বুঝবো:

- Relative path
- Absolute path
- Flutter এ ফাইল সাধারণত app-specific directory তে থাকে

ভুল path দিলে error হয়, তাই এটা খুব গুরুত্বপূর্ণ।

⚠️ 7. Exception কী?

Exception হলো runtime error, যেটা কোড চলার সময় ঘটে।

যেমন:

- ফাইল না পাওয়া
- পারমিশন না থাকা
- ভুল path

Exception না ধরলে অ্যাপ crash করতে পারে।

🧪 8. try–catch দিয়ে Exception Handling

আমরা শিখবো:

- try → ঝুঁকিপূর্ণ কোড
- catch → error ধরার জায়গা
- finally → যেটা সবসময় চলবে

এতে অ্যাপ:

- ক্র্যাশ করে না
- ইউজার-friendly থাকে

🎯 9. on বনাম catch

- catch → সব ধরনের error ধরতে
- on → নির্দিষ্ট exception ধরতে

এতে করে আমরা error-এর উপর বেশি control পাই।

🧩 10. Custom Exception তৈরি করা

নিজেদের প্রয়োজন অনুযায়ী আমরা:

- custom exception class তৈরি করতে পারি
- নির্দিষ্ট condition এ error throw করতে পারি

এটা বড় অ্যাপে খুব কাজে লাগে।

🔄 11. Asynchronous Error Handling

File operations সাধারণত async হয়।

আমরা শিখবো:

- async / await
- Future থেকে error ধরার উপায়

Flutter অ্যাপে এটা খুব গুরুত্বপূর্ণ।

Any questions? Ask away!

Give your audience space to resolve doubts or concerns. Open up the discussion before ending the meeting.

Thank you