10 Dart Golden Rules — 60 Seconds Mein

The 10 rules that keep Dart code clean and crash-free — one rapid revision card for the whole Dart series.

Introduction

This is the revision card for the whole Dart series — the ten rules from Lessons D.2 to D.9 squeezed onto one page. Read it before an exam, before an interview, or before you start the Flutter lessons: if these ten are in your fingers, your Dart is solid. Every rule carries the lesson it came from, so a weak rule tells you exactly which lesson to revisit. *(Roman Urdu: Yeh das usool poori Dart ka nichor hain — aik minute mein poori series revise.)*

Key points

1. **Har program `main()` se shuru hota hai** (D.2) — no main, no run. Dart enters through that one door.
2. **Har statement ke aakhir mein semicolon `;`** (D.2) — the missing `;` is the most common beginner error; check it first whenever Dart complains.
3. **Type pehle, value baad mein** (D.3) — decide the box's type honestly: `int`, `double`, `String`, `bool`. A box never changes its type.
4. **Naam wazeh rakho — lowerCamelCase** (D.3) — `studentName`, `totalMarks`. Future-you reads this code; `x` and `abc` are debts.
5. **Sentence mein value chahiye to `$` interpolation** (D.3) — `print('Total: $total')`, not pieces joined with `+`.
6. **`final` set-once ke liye, `const` pehle-se-maloom ke liye** (D.3) — locking values that should never change prevents whole families of bugs.
7. **Sawaal mein `==`, rakhne mein `=`** (D.4) — one equal puts a value in the box; two equals ask a question. Read conditions aloud as questions.
8. **Loop mein update lazmi** (D.5) — `i++` or `i--` bhoolna matlab infinite loop. Har loop se poochho: "yeh rukega kaise?"
9. **Khali value ko pehle check karo** (D.9) — `?.` safe call, `??` fallback, if-check for certainty. `!` sirf jab pura yaqeen ho.
10. **Function chhota, aik kaam wala** (D.6) — one function, one job, a name that says the job. Long functions hide bugs; small ones expose them.

Code examples

Example 1 — Rules 1, 2, 4 and 5 in six lines

```
void main() {
  String studentName = 'Ali';
  int totalMarks = 177;

  print('$studentName ke total marks: $totalMarks');
}
```

What it does: One entry point (rule 1), semicolons everywhere (rule 2), names a stranger can understand (rule 4), and interpolation instead of `+` joining (rule 5). This tiny program is what "clean Dart" looks like at the smallest size.

What you will see: one line — `Ali ke total marks: 177`. If any of the four rules breaks, this program either fails to run (rules 1–2) or becomes hard to read at a glance (rules 4–5).

Example 2 — Rules 6 and 7: locked values, honest questions

```
void main() {
  const int passingMarks = 50;
  final int myMarks = 82;

  if (myMarks == passingMarks) {
    print('Bilkul pass line par');
  } else if (myMarks > passingMarks) {
    print('Pass — aaram se');
  } else {
    print('Fail');
  }
}
```

What it does: The pass mark is a `const` — known before the program runs and impossible to change by accident (rule 6). The condition uses `==` to *ask* (rule 7); a single `=` here would be an assignment, and Dart would refuse it inside the question.

What you will see: one line — `Pass — aaram se`, because 82 is greater than 50. Try to add `passingMarks = 40;` and Dart stops you — the lock from rule 6 working in public.

Example 3 — Rule 9 + 10: a small function that can never crash

```
String gradeOf(int? marks) {
  if (marks == null) {
    return 'Marks maujood nahi';
  }
  return marks >= 50 ? 'Pass' : 'Fail';
}

void main() {
  print(gradeOf(82));
  print(gradeOf(null));
}
```

What it does: The function takes an honestly-nullable input and checks it before any use (rule 9), so *no caller can crash it*. And the function does exactly one job — marks in, verdict out (rule 10). Notice it needs no `!` anywhere: the if-check already promoted `marks` to safe.

What you will see: two lines — `Pass` and `Marks maujood nahi`. This pattern — check first, then trust — is the beating heart of the mini project in the next lesson.

Quick recap

- Start at `main()`, end statements with `;`.
- Types honest, names readable, values in sentences via `$`.
- `final` and `const` lock what should not change.
- Questions use `==`. Loops must update or they never stop.
- Null is handled before it is used — `?.`, `??`, if-check. Small functions, one job each.

Common mistakes

1. **Treating the rules as exam theory, not typing habits** — the rules only pay when your fingers follow them without thinking: semicolon while writing the line, camelCase while naming the box, check while touching a nullable.
2. **Reaching for `!` to end an argument with the compiler** — rule 9 exists because `!` converts a typing-time error into a run-time crash. Every `!` in your code should make you pause and justify it.
3. **One giant main()** — rule 10 is the one beginners break last and regret most. The moment a block of steps repeats or earns a name ("calculate average"), it deserves to become a function — D.11 shows the difference it makes.

Practice

1. Take any program you wrote for Lessons D.3–D.7 and audit it against all 10 rules, one by one. Write down which rule it breaks most — that is your personal weak rule.
2. From memory, write the table-of-5 program following every rule: clear names, interpolation, a `printTable` function that takes the number as a parameter, and a loop that provably stops.

Practice quiz

1. Golden rule 7: in a question inside a condition, which one do you write?

- =, one equal sign
- ==, two equal signs, because one equal puts a value in the box
- ===, three equal signs
- !=, always

Answer: ==, two equal signs, because one equal puts a value in the box

2. Golden rule 8: what must every loop have?

- A print statement
- An update like i++ or i--, or it never stops
- A break statement
- A list inside it

Answer: An update like i++ or i--, or it never stops

3. Golden rule 10: how should a function be shaped?

- One giant function for the whole program
- Small, one job each, with a name that says the job
- As long as possible, so it is impressive
- Without parameters, always

Answer: Small, one job each, with a name that says the job

Watch the video lesson: https://youtu.be/I5NEUHC-bNc

Read online: https://scopetreasure.com/notes/dart-golden-rules · More free tools and notes at https://scopetreasure.com