

## UVM PROGRAM WRITING CHECKLIST

//-----//

### Understand DUT First

- ✓ Interface signals
- ✓ Clock & reset behavior
- ✓ Input/output timing
- ✓ What to verify (functional intent)

Never start UVM without DUT clarity.

//-----//

### 1. Create Interface

- ✓ Define all DUT signals
- ✓ Add clocking block (optional but recommended)
- ✓ Use modports (driver / monitor)

```
interface dut_if(input logic clk);
```

//-----//

### 2. Transaction (uvm\_sequence\_item)

- ✓ Extend uvm\_sequence\_item
- ✓ Declare stimulus fields as rand
- ✓ Add constraints if required
- ✓ Register with factory

#### Checklist:

- ✓ rand variables
- ✓ new() constructor
- ✓ uvm\_object\_utils

//-----//

### 3. Sequence (uvm\_sequence)

- ✓ Extend uvm\_sequence
- ✓ Create transaction
- ✓ Randomize
- ✓ Start & finish item()

#### Checklist:

- ✓ start\_item()
- ✓ randomize()

- ✓ finish\_item()

//-----//

#### 4. Sequencer

- ✓ Declare as transaction\_sequencer type
- ✓ No custom logic needed initially

##### Checklist:

- ✓ Correct transaction type
- ✓ Factory registration

//-----//

#### 5. Driver

- ✓ Extend uvm\_driver
- ✓ Get transaction from sequencer
- ✓ Drive DUT signals
- ✓ Use clock synchronization

##### Checklist:

- ✓ seq\_item\_port.get\_next\_item()
- ✓ Drive signals
- ✓ item\_done()
- ✓ Reset handling

//-----//

#### 6. Monitor

- ✓ Extend uvm\_monitor
- ✓ Sample DUT signals
- ✓ Create transaction from observed data
- ✓ Send to scoreboard using analysis port

##### Checklist:

- ✓ uvm\_analysis\_port
- ✓ Sample at correct clock edge
- ✓ No driving here

//-----//

#### 7. Scoreboard

- ✓ Extend uvm\_scoreboard

- ✓ Receive transactions
- ✓ Implement reference model
- ✓ Compare expected vs actual
- ✓ Report PASS/FAIL

**Checklist:**

- ✓ uvm\_analysis\_imp
- ✓ Clear comparison logic
- ✓ uvm\_error on mismatch

//-----//

## 8. Agent

Instantiate:

- ✓ Driver
- ✓ Sequencer
- ✓ Monitor

Configure:

- ✓ Active / Passive

Checklist:

- ✓ Correct connections
- ✓ Driver ↔ Sequencer connected
- ✓ Factory registration for agents

//-----//

## 9. Environment

- ✓ Instantiate agents
- ✓ Instantiate scoreboard
- ✓ Connect monitor → scoreboard

**Checklist:**

- ✓ All components created
- ✓ Analysis connections done
- ✓ Optional: coverage collector

//-----//

## 10. Test

- ✓ Extend uvm\_test
- ✓ Create environment
- ✓ Start sequence
- ✓ Control simulation end

**Checklist:**

- ✓ raise\_objection
- ✓ Start sequence
- ✓ drop\_objection

//-----//

## 11. Top Module

- ✓ Instantiate DUT
- ✓ Instantiate interface
- ✓ Set interface via uvm\_config\_db
- ✓ Call run\_test()

**Checklist:**

- ✓ Virtual interface set
- ✓ run\_test() called

//-----//

## 12. Reporting & Debug

Use:

- ✓ uvm\_info
- ✓ uvm\_warning
- ✓ uvm\_error
- ✓ uvm\_fatal

**Checklist:**

- ✓ Meaningful TAG names
- ✓ No excessive prints

//-----//

## 11. Final Sanity Checks

- ✓ Simulation runs without fatal error
- ✓ Sequence executed
- ✓ Monitor captured data
- ✓ Scoreboard reported result

- ✓ Coverage goals met
- ✓ Factory overrides work

//-----//

## QUICK FLOW MEMORY LINE

**Transaction → Sequence → Sequencer → Driver → DUT → Monitor → Scoreboard**