

Chapter 1: Safety First



Chapter 2: Setting Up Your Playground



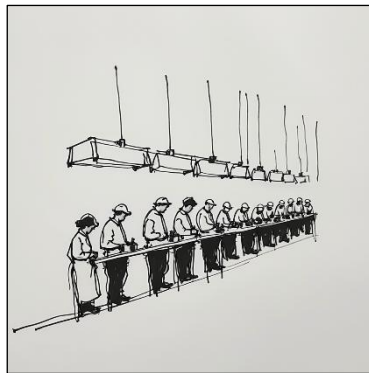
```
533     const text1b = text1[@intCast(usize, x)..];
534     const text2b = text2[@intCast(usize, y)..];
535
536     // Compute both diffs serially.
537     var diffs = try dmp.diffInternal(allocator, text1a, text2a, false, deadline);
538     var diffsb = try dmp.diffInternal(allocator, text1b, text2b, false, deadline);
539     defer diffsb.deinit(allocator);
540
541     try diffs.appendSlice(allocator, diffsb.items);
542     std.atomic.|
543     return diff
544 }
545
546 /// Do a quick
547 /// greater acc
548 /// This speedu
549 /// @param text
550 /// @param text
551 /// @param deadline Time when the diff should be complete by.
552 /// @return List of Diff objects.
553 fn diffLineMode(
```

cache_line	switch (builtin.cpu.arch)
Ordering	std.builtin.AtomicOrder
compilerFence(comptime ordering: Orderi...	vo...
fence(comptime ordering: Ordering)	void arts for
spinLoopHint()	void
Atomic(comptime T: type)	type
Queue(comptime T: type)	type
Stack(comptime T: type)	type

Chapter 3: Your First Zig Program



Chapter 4: Control Flow

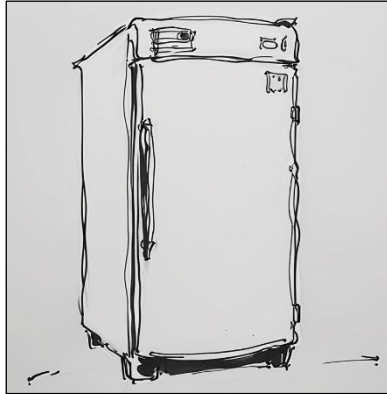


Chapter 5: Functions for the Efficient Programmer

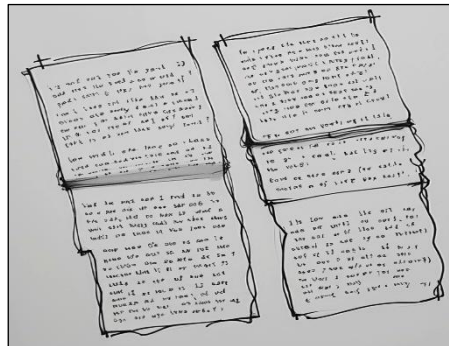
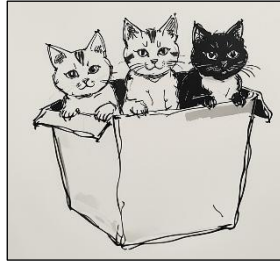




Chapter 6: Error Handling

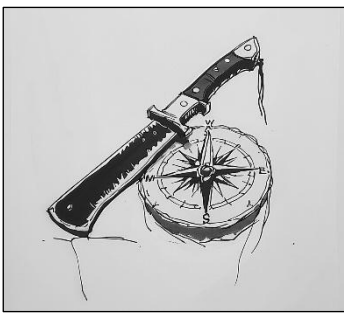


Chapter 7: Testing Your Zig Code



Chapter 10: The Standard Library





Chapter 13: Real-World Zig

