

Lessons Learned: Scoring System Investigation

Executive Summary

Attempted to fix scoring issues but introduced regressions. Need to restore to v2.1 and restart with minimal, targeted approach.

Current State Assessment (Before Restore)

- **Working**: Auto-arrange system, arrangement generation, basic gameplay
- **Broken**: Scoring accuracy, test framework alignment, naming consistency
- **Tests**: 13/17 passing (down from previous)

Key Discoveries

1. Architecture Understanding

- **HandDetector**: Creates base hands + stores 3 position scores per hand
- **ScoringUtilities**: Calculates points based on hand type + position
- **BestArrangementGenerator**: Uses `hand.positionScores.front/middle/back`
- **Architecture is sound** - no need to change core structure

2. Naming Inconsistencies Found

Component	Four of a Kind	Five of a Kind	Large Hands
HandDetector	`"4 of a Kind"`	`"5 of a Kind"`	`"6-card Straight Flush"`
card-evaluation.js	`"Four of a Kind"`	`"Five of a Kind"`	Various
ScoringUtilities	`"four of a kind"`	`"five of a kind"`	Lowercase

Impact: Scoring lookups fail due to name mismatches

3. Scoring Issues Identified

Issue	Expected	Actual	Root Cause
Three of a Kind (front)	3	1	Condition order
Straight Flush (front)	15	4	Condition order
Multiple scores	Single	"4,15"	Overlapping detection
Four of a Kind	13 hands	1 hand	Test expectation wrong

4. Technical Debt Identified

- **Naming inconsistency** across components
- **Dual format support** needed as bridge solution
- **Overlapping detection** in HandDetector creating duplicate hands
- **Test framework** expectations don't match HandDetector reality

What Went Wrong

1. Changed Too Much at Once

- Modified HandDetector, ScoringUtilities, and test framework simultaneously
- Lost track of which changes caused which effects
- Introduced regressions in working systems

2. Misunderstood 4K Expansion

- Assumed HandDetector should create 13 "4K + kicker" combinations
- **Reality**: HandDetector creates base hands, BestArrangementGenerator adds kickers
- Test framework expectations were incorrect

3. Condition Order Bug in ScoringUtilities

```
```javascript
// WRONG ORDER:
if (handName.includes('flush')) return 4; // Catches "straight flush"
if (handName.includes('straight flush')) return 15; // Never reached

// CORRECT ORDER:
if (handName.includes('straight flush')) return 15; // Check specific first
if (handName.includes('flush')) return 4; // Then general
```
```

4. Overlapping Detection Issue ❌

- Same cards being detected as both "Straight" and "Straight Flush"
- Causes multiple position scores for same hand type
- Test audit shows "4,15" instead of single values

Proper Fix Strategy (Restart Plan)

Phase 1: Minimal Scoring Fixes 🌀

****Goal**:** Fix only the specific scoring bugs without changing architecture

1. ****Restore to v2.1**** first
2. ****Fix ScoringUtilities condition order**** (single change)
 - Test immediately with single test case
 - Commit if successful
3. ****Add dual naming support**** (single change)
 - Support both "Four of a Kind" and "4 of a Kind"
 - Test immediately
 - Commit if successful
4. ****Fix Three of a Kind front scoring**** (single change)

Phase 2: Clean Detection Logic ✂️

****Goal**:** Fix overlapping detection without breaking existing functionality

1. ****Identify overlap patterns**** in HandDetector
2. ****Fix one detection method at a time****
3. ****Extensive testing after each change****

Phase 3: Naming Standardization 🗂️

****Goal**:** Consistent naming across all components (after everything works)

1. ****Choose standard**** (recommend HandDetector format)
2. ****Update one component at a time****
3. ****Remove dual format support**** once standardized

Testing Strategy

Before Any Changes:

- Run full test suite to establish baseline
- Document exactly what works and what doesn't
- Identify minimal reproducible test cases

During Changes:

- ****One change at a time****
- Test immediately after each change
- Rollback if any regression detected
- Commit working changes before proceeding

Validation Criteria:

- All tests must pass or maintain previous pass rate
- No regressions in working functionality
- Auto-arrange system continues to work
- Game remains playable

Commit Strategy

```
```bash
```

```
Safe branching approach
```

```
git checkout v2.1
```

```
git checkout -b fix-scoring-minimal
```

```
Make one small change
```

```
Test extensively
```

```
Commit if successful
git add specific-file.js
git commit -m "Fix: Single specific issue - verified working"
```





```
Repeat for next issue
````
```

Key Principles Moving Forward





1. **Minimal Changes**: Never change more than one thing at a time
2. **Test Immediately**: Every change gets tested before proceeding
3. **Preserve Working Code**: If something works, don't break it
4. **Document Assumptions**: Verify understanding before making changes
5. **Rollback Readily**: Better to go back than push forward broken code

Success Metrics

Must Maintain:

-  Auto-arrange functionality working
-  Game playability preserved
-  Test pass rate maintained or improved
-  No performance regressions

Target Improvements:

-  Three of a Kind front: 1 → 3 points
-  Straight Flush front: 4 → 15 points
-  Consistent naming across components
-  Single scores instead of multiple values

Files That Need Careful Attention

| File | Risk Level | Changes Needed |
|---------------------------------|------------|------------------------|
| ----- ----- ----- | | |
| `js/hands/scoring-utilities.js` | Medium | Condition order fix |
| `js/hands/hand-detector.js` | High | Overlapping detection |
| `js/hands/card-evaluation.js` | Low | Naming standardization |
| `js/tests/*` | Medium | Expectation alignment |

Next Action: Restore to v2.1 and restart with Phase 1, single issue at a time.