BBTTCC Radiation v4.8.1-ENHANCED - Updated Testing Guide

Overview

This testing guide covers the enhanced BBTTCC Radiation module (v4.8.1-ENHANCED) designed for FoundryVTT v13+ and D&D5e v5.1.4+ compatibility. The module provides comprehensive radiation tracking, environmental hazard management, and mutation-like effects.

System Requirements

- **FoundryVTT**: v13.0+ (tested on v13.348)
- **D&D5e System**: v5.1.4+ (tested on v5.1.4)
- **Module Dependencies**: None (standalone module)

Module Installation

1. Copy folder to FoundryVTT `Data/modules/` directory
2. Rename folder to exactly `bbttcc-radiation` (no version suffix)
3. Enable module in FoundryVTT Module Management
4. Refresh FoundryVTT

Test Case 1: Installation & Module Initialization

Core Functionality Test

Expected Behavior:

- ✅ Module loads without critical errors
- ✅ Modern Application patterns for FoundryVTT v13+
- ✅ Flags-based data storage system
- ✅ Async/await patterns throughout
- ✅ Settings registration and API exposure

Installation Verification

```javascript
```

```
// Console test - check module status
const radiationMod = game.modules.get('bbttcc-radiation');
console.log('Module found:', !!radiationMod);
console.log('Module active:', radiationMod?.active);
console.log('API available:', !!radiationMod?.api);

// Check system compatibility
console.log('Game version:', game.version);
console.log('System:', game.system.id, game.system.version);
```

Expected Results

- Console shows: "BBTTCC Radiation v4.8.1-ENHANCED | System fully operational"
- No JavaScript errors in console
- Settings menu appears under "Configure Settings" → "Module Settings"
- API methods available via `game.modules.get('bbttcc-radiation').api`

Test Case 2: Settings Configuration

Settings Registration Test

Objective: Verify all module settings are properly registered

Settings to Verify

1. **Enable Automatic Tracking** (Boolean, default: true)
2. **Tracking Interval** (Number, default: 180 seconds)
3. **Show Token HUD Controls** (Boolean, default: true)
4. **Enable Radiation Decay** (Boolean, default: true)
5. **Debug Mode** (Boolean, default: false)
6. **Default Zone Type** (String, default: 'background')

Test Steps

1. Navigate to Configure Settings → Module Settings
2. Locate "BBTTCC Radiation v4.8.1-ENHANCED" section
3. Verify all 6 settings are present and functional
4. Test changing values and saving

5. Reload world and verify settings persist

Test Case 3: Actor Flag-Based Data Integration

Radiation Data Structure Test

Objective: Verify radiation data integrates with D&D5e actors using flags

Test Steps

```javascript
// Create or select a test character
const testActor = game.actors.getName("Test Character") ||
  await Actor.create({
    name: "Radiation Test Subject",
    type: "character",
    system: {}
  });

// Check if radiation data initializes
const radiationAPI = game.modules.get('bbttcc-radiation').api;
const radiationData = radiationAPI.getRadiationData(testActor.token);
console.log('Radiation data:', radiationData);

// Verify flag structure
const flagData = testActor.getFlag('bbttcc-radiation', 'radiation');
console.log('Flag data:', flagData);
```

Expected Data Structure

```javascript
```

```
{
  level: 0,         // Current radiation level (0-100%)
  exposure: 0,      // Total exposure points
  threshold: 100,   // Threshold for effects
  effects: [],      // Applied effect IDs
  protection: 0,    // Protection percentage
  lastUpdate: null, // ISO timestamp
  accumulation: 'linear',
  decay: {
    enabled: true,
    rate: 1,
    interval: 3600
  }
}
```

Test Case 4: Radiation Application & Tracking

Manual Radiation Application

Objective: Test core radiation exposure mechanics

API Testing

```javascript
const api = game.modules.get('bbttcc-radiation').api;
const token = canvas.tokens.controlled[0]; // Select a token first

if (token) {
  // Test radiation application
  await api.updateRadiationExposure(token, 25, { notify: true });
  console.log('Applied 25 radiation points');

  // Check updated data
  const updated = api.getRadiationData(token);
  console.log('New level:', updated.level);
  console.log('Effective level:', updated.effectiveLevel);
  console.log('Radiation level category:', updated.radiationLevel.name);
}
```

Protection Testing

```javascript
// Set protection level
await api.setProtectionLevel(token, 'hazmat'); // 30% protection

// Apply radiation with protection
await api.updateRadiationExposure(token, 20, { notify: true });

// Verify protection reduced effective exposure
const protectedData = api.getRadiationData(token);
console.log('Protection effectiveness:', protectedData.protection);
```

Expected Results

- Chat notifications appear for exposure changes
- Protection correctly reduces effective exposure
- Radiation level categories update (Safe → Low → Moderate → High → Severe → Lethal)
- Data persists between sessions

Test Case 5: Scene-Based Radiation Zones

Zone Configuration Test

Objective: Test environmental radiation zone system

Zone Setup

```javascript
```

```javascript
const api = game.modules.get('bbttcc-radiation').api;

// Set scene radiation zone
await api.setSceneRadiationZone(canvas.scene, 'industrial', 25);

// Verify zone data
const zoneData = api.getSceneRadiationZone(canvas.scene);
console.log('Zone type:', zoneData.type);
console.log('Intensity:', zoneData.intensity);
console.log('Description:', zoneData.description);

// Test available zone types
console.log('Available zones:', Object.keys(api.ZONE_TYPES));
```

Zone Types to Test

- `background` (Intensity: 1) - Natural background radiation
- `urban` (Intensity: 5) - Post-apocalyptic urban environment
- `industrial` (Intensity: 15) - Contaminated industrial areas
- `military` (Intensity: 25) - Former military installations
- `reactor` (Intensity: 40) - Nuclear facility areas
- `ground_zero` (Intensity: 60) - Direct bomb impact sites
- `hot_zone` (Intensity: 80) - Extreme contamination areas

Test Case 6: Automatic Radiation Tracking

Tracking System Test

Objective: Verify automatic radiation accumulation over time

Enable Tracking

```javascript
```

```javascript
const api = game.modules.get('bbttcc-radiation').api;

// Verify tracking is enabled
console.log('Auto tracking enabled:',
  game.settings.get('bbttcc-radiation', 'enableAutomaticTracking'));

// Check tracking interval
console.log('Tracking interval:',
  game.settings.get('bbttcc-radiation', 'trackingInterval'), 'seconds');

// Force a radiation tick for testing
await BBTTCCRadiationModule.processRadiationTick();
```

Expected Behavior

- Tokens in contaminated scenes accumulate radiation over time
- Higher intensity zones cause faster accumulation
- Protection reduces accumulation rate
- Background radiation allows natural decay

Test Case 7: Radiation Effects System

Active Effects Integration

Objective: Test radiation effects application to characters

Effect Application Test

```javascript
```

```javascript
const api = game.modules.get('bbttcc-radiation').api;
const token = canvas.tokens.controlled[0];

// Apply high radiation to trigger effects
await api.updateRadiationExposure(token, 75, { notify: true });

// Check applied effects
const actor = token.actor;
const radiationEffects = actor.effects.filter(e =>
  e.flags['bbttcc-radiation']?.radiationEffect
);

console.log('Radiation effects applied:', radiationEffects.length);
radiationEffects.forEach(effect => {
  console.log('- Effect:', effect.name);
});
```

Effect Categories to Test

- **Safe (0-10%)**: No effects
- **Low (11-25%)**: Mild Radiation Sickness
- **Moderate (26-50%)**: Moderate Radiation Sickness, Fatigue
- **High (51-75%)**: Severe Radiation Sickness, Exhaustion Level 1
- **Severe (76-90%)**: Radiation Poisoning, Exhaustion Level 2, Poison Vulnerability
- **Lethal (91-100%)**: Severe Radiation Poisoning, Exhaustion Level 3, Partial Paralysis

---

Test Case 8: UI Applications Testing

Radiation Tracker Interface

Objective: Test the RadiationTracker Application

Open Radiation Tracker

```javascript
```

```javascript
const api = game.modules.get('bbttcc-radiation').api;
const token = canvas.tokens.controlled[0];

if (token) {
  // Open radiation tracker
  api.openRadiationTracker(token);

  // Alternative method
  const tracker = new api.RadiationTracker(token);
  tracker.render(true);
}
```

Tracker Features to Test

- Current radiation status display
- Manual exposure adjustment buttons (+/- 1, 5, 10)
- Protection type selection dropdown
- Custom protection percentage input
- Reset radiation functionality
- Remove effects functionality
- Real-time data updates

Zone Configuration Interface

```javascript
// Open zone configuration
api.openZoneConfig(canvas.scene);

// Alternative method
const zoneConfig = new api.RadiationZoneConfig(canvas.scene);
zoneConfig.render(true);
```

Zone Config Features to Test

- Scene statistics display
- Zone type selection
- Custom intensity override
- Effects preview

- Batch operations (reset all, apply protection)
- Token effects preview

Test Case 9: Token HUD Integration

HUD Controls Test

Objective: Verify radiation controls appear in Token HUD

Prerequisites

- Setting "Show Token HUD Controls" must be enabled
- User must be GM
- Token must have an associated actor

Test Steps

1. Select a token with radiation data
2. Right-click to open Token HUD
3. Look for radiation icon in HUD controls
4. Click radiation icon to open tracker

Expected Results

- Radiation icon appears with color coding based on level
- Icon shows current radiation level in tooltip
- Clicking opens RadiationTracker application

Test Case 10: Macro Integration

Modern Macro Test

Objective: Test the included CreateBBTTCCRadiationZone-MODERN.js macro

Macro Execution

1. Import the macro from the module's macros folder
2. Execute the macro as GM
3. Verify enhanced dialog appears with:

- Current scene statistics
- Zone type selection with descriptions
- Custom intensity options
- Advanced configuration options

Macro Features to Test

- Scene statistics (token count, affected tokens)
- Zone type descriptions
- Custom intensity override
- Advanced configuration integration
- Error handling and user feedback

Test Case 11: Error Handling & Edge Cases

Error Conditions to Test

Invalid Data Handling

```javascript
const api = game.modules.get('bbttcc-radiation').api;

// Test with null/undefined token
try {
  await api.updateRadiationExposure(null, 10);
} catch (error) {
  console.log('Properly caught error:', error.message);
}

// Test with invalid protection type
try {
  await api.setProtectionLevel(token, 'invalid_type');
} catch (error) {
  console.log('Properly caught error:', error.message);
}
```

Boundary Testing

- Apply negative radiation values (should be prevented/clamped)

- Apply radiation exceeding 100% (should be clamped)
- Set protection values outside 0-100% range
- Test with actors that have no token representation

Test Case 12: Performance & Data Persistence

Performance Testing

```javascript
// Test bulk operations
const tokens = canvas.tokens.placeables.slice(0, 10);
const startTime = performance.now();

for (const token of tokens) {
  if (token.actor) {
    await api.updateRadiationExposure(token, 5);
  }
}

const endTime = performance.now();
console.log(`Processed ${tokens.length} tokens in ${endTime - startTime}ms`);
```

Data Persistence Testing

1. Apply radiation to multiple tokens
2. Save and reload the world
3. Verify all radiation data persists correctly
4. Check that tracking resumes properly

Integration Testing with Other Modules

BBTTCC Module Compatibility

If other BBTTCC modules are present:

```javascript
```

```
// Check for other BBTTCC modules
const bbttccModules = game.modules.filter(m =>
  m.id.includes('bbttcc') && m.active
);
console.log('Active BBTTCC modules:', bbttccModules.map(m => m.id));

// Test cross-module integration if available
const factionsMod = game.modules.get('bbttcc-factions');
if (factionsMod?.active) {
  console.log('Factions integration available');
  // Test faction-based radiation mechanics
}
```

Expected Success Criteria

Module Must:

- ✅ Load without console errors in FoundryVTT v13.348
- ✅ Integrate seamlessly with D&D5e v5.1.4+
- ✅ Persist data reliably using flags-based storage
- ✅ Provide functional UI applications (RadiationTracker, RadiationZoneConfig)
- ✅ Handle errors gracefully with user-friendly messages
- ✅ Support automatic radiation tracking with configurable intervals
- ✅ Apply and remove Active Effects correctly
- ✅ Integrate with Token HUD (when enabled)

Performance Benchmarks

- Zone configuration: < 2 seconds for complex scenes
- Individual radiation updates: < 100ms per token
- Batch operations: 10+ tokens processed in < 5 seconds
- Memory usage: Stable with automatic tracking enabled

Common Issues & Solutions

1. "Module not found" Errors

Cause: Incorrect folder naming **Solution**: Ensure folder is named exactly `bbttcc-radiation`

2. "API not available" Errors

Cause: Module not fully initialized **Solution**: Use `await api.waitForReady()` before API calls

3. Data Not Persisting

Cause: Flag update failures **Solution**: Check actor permissions and flag structure

4. Token HUD Not Showing Controls

Cause: Setting disabled or user not GM **Solution**: Enable "Show Token HUD Controls" setting

5. Automatic Tracking Not Working

Cause: Setting disabled or no scene radiation **Solution**: Enable automatic tracking and set scene radiation zone

---

Diagnostic Commands

Quick Status Check

```javascript
// Comprehensive module status
const mod = game.modules.get('bbttcc-radiation');
console.log({
  found: !!mod,
  active: mod?.active,
  hasAPI: !!mod?.api,
  isReady: mod?.api?.isReady?.() || false,
  settingsRegistered: !!game.settings.settings.get('bbttcc-radiation.enableAutomaticTracking')
});
```

Scene Radiation Status

```javascript
```

```
const api = game.modules.get('bbttcc-radiation').api;
if (api && canvas.scene) {
  const zone = api.getSceneRadiationZone();
  console.log('Scene radiation:', zone);

  const tokens = canvas.tokens.placeables.filter(t => t.actor);
  console.log(`Tokens in scene: ${tokens.length}`);

  tokens.forEach(token => {
    const data = api.getRadiationData(token);
    if (data) {
      console.log(`${token.name}: ${data.radiationLevel.name} (${data.level}%)`);
    }
  });
}
```

Updated for BBTTCC Radiation v4.8.1-ENHANCED

Target Environment: FoundryVTT v13.348, D&D5e v5.1.4+