ATLASENGINE COMPLETE GUIDE - PART 4

Game Mechanics, Combat, Inventory, Complete Examples

CATEGORY 8: GAME MECHANICS - COMBAT (60 COMMANDS)

Player Stats

1. health / heal

Purpose: Manage player health **Syntax:** `health is value` or `heal amount` **Examples:**

```
# Set health
health is 100

# Take damage
health subtract 25          → health = 75

# Heal
heal 20                     → health = 95

# Max health cap
when health greater 100 {
    health is 100
}

# Death check
when health less 1 {
    say "Game Over"
    gameover
}
```

---

2. mana

Purpose: Magic/ability resource **Syntax:** `mana is value`

```
mana is 100
```

```
# Cast spell
when mana greater 20 {
    mana subtract 20
    cast "fireball"
}

# Regenerate
mana add 1
when mana greater 100 {
    mana is 100
}
```

---

3. stamina

Purpose: Sprint/action resource **Syntax:** `stamina is value`

```
stamina is 100

# Sprint
when sprint_key equals 1 {
    when stamina greater 0 {
        speed is 10
        stamina subtract 1
    }
}

# Regenerate
when not_sprinting equals 1 {
    stamina add 2
}
```

---

4. armor / defense

Purpose: Damage reduction **Syntax:** `armor is value`

```
armor is 50

# Take damage with armor
damage_taken is 100
actual_damage is damage_taken - armor
when actual_damage less 0 {
    actual_damage is 0
}
```

```
health subtract actual_damage
```

---

5. attack / accuracy

Purpose: Attack power/hit chance **Syntax:** `attack is value` or `accuracy is value`

```
attack is 25
accuracy is 80  # 80% hit chance

# Attack calculation
hit_chance is random 1 100
when hit_chance less accuracy {
    # Hit!
    damage is attack
    enemy_health subtract damage
}
```

---

6. critical

Purpose: Critical hit chance/damage **Syntax:** `critical chance multiplier`

```
crit_chance is 20  # 20% chance
crit_mult is 2.0   # 2x damage

roll is random 1 100
when roll less crit_chance {
    # Critical hit!
    damage multiply crit_mult
    shout "CRITICAL HIT!"
}
```

---

7. dodge / parry / block

Purpose: Defense actions **Syntax:** `dodge chance` or `parry` or `block`

```
# Dodge
dodge_chance is 30
roll is random 1 100
when roll less dodge_chance {
    say "Dodged!"
    damage is 0
```

```
}

# Block
when block_active equals 1 {
    damage multiply 0.5  # Half damage
}

# Parry
when parry_timing equals 1 {
    damage is 0
    counter_attack is 1
}
```

Weapons

8. weapon

Purpose: Define weapon **Syntax:** `weapon name damage firerate ammo`

```
# Pistol
weapon "pistol" damage 25 firerate 0.5 ammo 12

# Shotgun
weapon "shotgun" damage 50 firerate 1.0 ammo 8

# Rifle
weapon "rifle" damage 30 firerate 0.3 ammo 30
```

---

9. equip / unequip

Purpose: Equip/unequip weapon **Syntax:** `equip weapon` or `unequip weapon`

```
equip "pistol"
say "Pistol equipped"

unequip "pistol"
equip "shotgun"
say "Switched to shotgun"
```

---

10. shoot / fire / cast

Purpose: Use weapon/spell **Syntax:** `shoot` or `fire weapon` or `cast spell`

```
# Shoot
when mouse_click equals 1 {
    when ammo greater 0 {
        shoot
        ammo subtract 1
    }
}

# Cast spell
when key_q equals 1 {
    when mana greater 30 {
        cast "fireball"
        mana subtract 30
    }
}
```

---

11. reload

Purpose: Reload weapon **Syntax:** `reload weapon`

```
when key_r equals 1 {
    when magazine less max_ammo {
        reload "pistol"
        say "Reloading..."
        # Wait 2 seconds
        magazine is max_ammo
    }
}
```

---

12. ammo / magazine

Purpose: Ammo management **Syntax:** `ammo is value` or `magazine is value`

```
ammo is 30        # Total ammo
magazine is 12    # Current magazine
max_ammo is 12    # Magazine capacity

# Fire
magazine subtract 1
when magazine equals 0 {
    # Auto reload
    reload "pistol"
}
```

---

13. spread

Purpose: Weapon accuracy/spread **Syntax:** `spread weapon value`

```
spread "pistol" 2     # Low spread (accurate)
spread "shotgun" 15   # High spread (inaccurate)

# Apply spread to shot
bullet_x add random -spread spread
bullet_y add random -spread spread
```

---

14. recoil

Purpose: Weapon kickback **Syntax:** `recoil amount`

```
recoil 5
# Camera moves up by 5

# Recoil pattern
when shooting equals 1 {
    camera_y add recoil
    recoil add 2  # Increases with sustained fire
}
```

---

15. firerate

Purpose: Rate of fire **Syntax:** `firerate weapon value`

```
firerate "pistol" 0.5    # 0.5 seconds between shots
firerate "minigun" 0.05  # 0.05 seconds (20 shots/sec)

# Cooldown
when can_shoot equals 1 {
    shoot
    can_shoot is 0
    cooldown is firerate
}
```

---

16. scope / aim / zoom

Purpose: Aiming mechanics **Syntax:** `scope zoom` or `aim` or `zoom level`

```
# Scope
when right_click equals 1 {
    scope 2.0  # 2x zoom
    sensitivity multiply 0.5
}

# Aim down sights
when aim_active equals 1 {
    accuracy add 20
    speed multiply 0.7
}
```

Weapon Types

17. pistol

Example pistol setup:

```
weapon "pistol" damage 25 firerate 0.5 ammo 12
accuracy is 85
spread is 2
recoil is 3
```

18. shotgun

Example shotgun setup:

```
weapon "shotgun" damage 50 firerate 1.2 ammo 8
accuracy is 60
spread is 15
recoil is 8
pellets is 8  # Multiple projectiles
```

19. rifle / sniper

Example rifle setup:

```
weapon "rifle" damage 75 firerate 1.0 ammo 5
accuracy is 95
spread is 0.5
recoil is 10
scope_zoom is 4.0
```

20. bow / arrow

Example bow setup:

```
weapon "bow" damage 40 firerate 1.5 ammo 20
accuracy is 90
gravity_enabled is 1  # Arrow drops
trajectory is "parabolic"
```

21. sword / melee

Example melee setup:

```
weapon "sword" damage 35 firerate 0.8
range is 2  # Short range
swing_angle is 90  # Attack arc
```

22. laser / gun

Example laser setup:

```
weapon "laser" damage 20 firerate 0.1 ammo 100
projectile_speed is instant
hit_scan is 1  # Instant hit
```

23. rocket / grenade / bomb

Example explosive setup:

```
weapon "rocket" damage 100 firerate 2.0 ammo 5
projectile_speed is 30
explode_radius is 10
explode_damage is 100
```

24. bullet / projectile

Create projectile:

```
# Bullet
bullet "bullet1" from player_x, player_y to target_x, target_y
velocity "bullet1" speed 50

# Projectile with gravity
projectile "arrow1" from bow_pos to target
gravity "arrow1" enable
```

Combat Actions

25. hit

Purpose: Register hit on target **Syntax:** `hit target damage`

```
when bullet_collide equals 1 {
    hit "enemy1" damage 25
    say "Hit enemy for 25 damage"
}
```

26. explode

Purpose: Create explosion **Syntax:** `explode at x, y, z radius r damage d`

```
explode at grenade_x, grenade_y, grenade_z radius 5 damage 50

# Damage all in radius
when distance less 5 {
    hit "enemy" damage 50
}
```

27. burn / poison / freeze / stun

Purpose: Status effects **Syntax:** `burn target duration` or `poison target dps duration`

```
# Burn (damage over time)
burn "enemy" duration 5
repeat 5 times {
    health subtract 10
    # Wait 1 second
}

# Poison
poison "player" dps 5 duration 10

# Freeze
freeze "enemy" duration 3
enemy_speed is 0

# Stun
stun "enemy" duration 2
enemy_can_move is 0
```

28. buff / debuff

Purpose: Stat modifications **Syntax:** `buff stat amount duration` or `debuff stat amount duration`

```
# Attack buff
buff attack 20 duration 10
attack add 20
```

```
# After 10 seconds
attack subtract 20

# Defense debuff
debuff armor 15 duration 5
armor subtract 15
```

29. enchant

Purpose: Add weapon effect **Syntax:** `enchant weapon effect`

```
enchant "sword" effect "fire"
# Sword now deals fire damage

enchant "bow" effect "poison"
# Arrows poison targets
```

30. cooldown

Purpose: Ability cooldown **Syntax:** `cooldown ability duration`

```
when ability_used equals 1 {
    cooldown "fireball" 5
    fireball_ready is 0
    # Wait 5 seconds
    fireball_ready is 1
}
```

Inventory System

31. inventory

Purpose: Player inventory **Syntax:** `inventory create size`

```
inventory create size 20
inventory_items is []
inventory_count is 0
```

32. additem / pickup

Purpose: Add item to inventory **Syntax:** `additem name` or `pickup item`

```
additem "health_potion"
inventory_items append "health_potion"
inventory_count add 1

say "Picked up Health Potion"
```

33. removeitem / dropitem

Purpose: Remove item from inventory **Syntax:** `removeitem name` or `dropitem name`

```
removeitem "health_potion"
inventory_items remove "health_potion"
inventory_count subtract 1

dropitem "sword" at player_x, player_y
```

34. useitem

Purpose: Use item from inventory **Syntax:** `useitem name`

```
when key_1 equals 1 {
    useitem "health_potion"
    when "health_potion" in inventory {
        heal 50
        removeitem "health_potion"
        say "Used Health Potion"
    }
}
```

35. hasitem

Purpose: Check if player has item **Syntax:** `hasitem name`

```
has_key is hasitem "key"
when has_key equals 1 {
    say "You can open the door"
} else {
    say "You need a key"
}
```

36. coin / gem / gold

Purpose: Currency **Syntax:** `gold add amount` or `gold subtract amount`

```
gold is 100

# Buy item
price is 50
when gold greater price {
    gold subtract price
    additem "sword"
    say "Purchased sword"
}
```

```
# Sell item
gold add 25
say "Sold item for 25 gold"
```

Leveling & Progression

37. level / levelup

Purpose: Player level **Syntax:** `level is value` or `levelup`

```
level is 1
exp is 0
exp_needed is 100

# Gain experience
exp add 50

# Level up check
when exp greater exp_needed {
    levelup
    level add 1
    exp subtract exp_needed
    exp_needed multiply 1.5

    shout "LEVEL UP!"
    health is 100
    attack add 5
}
```

38. xp / exp

Purpose: Experience points **Syntax:** `xp add amount`

```
# Kill enemy
when enemy_dead equals 1 {
    xp add 25
    say "+25 XP"
}
```

39. stat / ability / skill

Purpose: Character stats **Syntax:** `stat name value`

```
# Stats
strength is 10
agility is 8
intelligence is 12
```

```
# Level up stat
when levelup equals 1 {
    strength add 2
    agility add 1
}

# Skills
skill "swordfighting" level 5
skill "archery" level 3
```

40. powerup

Purpose: Temporary power boost **Syntax:** `powerup type duration`

```
powerup "speed" duration 10
speed multiply 2
# After 10 seconds
speed divide 2

powerup "invincibility" duration 5
invincible is 1
```

Quest System

41. quest / objective

Purpose: Quest management **Syntax:** `quest name objective description`

```
quest "main_quest" objective "Find the key"
quest_active is 1

# Quest progress
keys_found is 0
keys_needed is 5

when key_collected equals 1 {
    keys_found add 1
    say "Keys: " keys_found "/" keys_needed
}

when keys_found equals keys_needed {
    completequest "main_quest"
}
```

42. completequest

Purpose: Complete quest **Syntax:** `completequest name`

```
when objective_done equals 1 {
    completequest "main_quest"
    shout "Quest Complete!"
    reward "gold" 100
    xp add 500
}
```

43. reward

Purpose: Give quest reward **Syntax:** `reward type amount`

```
reward "gold" 100
reward "exp" 500
reward "item" "legendary_sword"
```

Enemy System

44. enemy / spawn

Purpose: Create enemy **Syntax:** `enemy name at x, y, z health h damage d`

```
enemy "goblin" at 10, 0, 10 health 50 damage 10
enemy "boss" at 20, 0, 20 health 500 damage 50

spawn "zombie" at 5, 0, 5
```

45. ai / behavior

Purpose: AI behavior **Syntax:** `ai enemy behavior type` **Types:** idle, patrol, chase, attack, flee

```
ai "goblin" behavior "patrol"
ai "wolf" behavior "chase"
ai "boss" behavior "attack"
```

46. chase / flee / attack

Purpose: AI actions **Syntax:** `chase target` or `flee from` or `attack target`

```
# Enemy AI
distance is distance player enemy

when distance less 10 {
    chase "player"
```

```
}

when distance less 2 {
    attack "player"
}

when enemy_health less 20 {
    flee from "player"
}
```

47. patrol / attract / repel

Purpose: Movement behaviors **Syntax:** `patrol waypoints` or `attract to point` or `repel from point`

```
# Patrol
waypoints is [[0,0,0], [10,0,0], [10,0,10], [0,0,10]]
patrol "guard" waypoints

# Attract to sound
attract "enemy" to noise_x, noise_y, noise_z

# Repel from danger
repel "enemy" from explosion_x, explosion_y, explosion_z
```

48. summon

Purpose: Summon creature/ally **Syntax:** `summon type at x, y, z`

```
when spell_cast equals 1 {
    summon "skeleton" at player_x, player_y, player_z
    summon_count add 1
}

# Allied unit
summon "knight" at 5, 0, 5
ai "knight" behavior "defend" target "player"
```

Game Flow

49. gameover / lose

Purpose: End game (lose condition) **Syntax:** `gameover` or `lose`

```
when health less 1 {
    gameover
    shout "YOU DIED"
```

```
    say "Final Score: " score
}

when lives equals 0 {
    lose
}
```

50. win

Purpose: End game (win condition) **Syntax:** `win`

```
when boss_dead equals 1 {
    win
    shout "VICTORY!"
    say "Congratulations!"
    say "Final Score: " score
}
```

51. checkpoint / respawn

Purpose: Save point **Syntax:** `checkpoint at x, y, z` or `respawn at x, y, z`

```
# Save checkpoint
when player_touch_checkpoint equals 1 {
    checkpoint at player_x, player_y, player_z
    say "Checkpoint saved"
}

# Death
when health less 1 {
    respawn at checkpoint_x, checkpoint_y, checkpoint_z
    health is 100
    lives subtract 1
}
```

52. lives

Purpose: Extra lives **Syntax:** `lives is value`

```
lives is 3

when health less 1 {
    lives subtract 1
    when lives greater 0 {
        respawn at checkpoint
        health is 100
    } else {
```

```
        gameover
    }
}
```

53. score / highscore

Purpose: Score tracking **Syntax:** `score add amount` or `highscore check`

```
score is 0

# Add points
when enemy_killed equals 1 {
    score add 100
}

# Check highscore
when game_over equals 1 {
    when score greater highscore {
        highscore is score
        say "New High Score!"
    }
}
```

54. timer / countdown

Purpose: Time limit **Syntax:** `timer start duration` or `countdown from time`

```
timer start 60  # 60 seconds

when timer equals 0 {
    gameover
    say "Time's up!"
}

# Count down
countdown from 10
say "9..."
say "8..."
```

55. pause / resume

Purpose: Pause game **Syntax:** `pause` or `resume`

```
when key_escape equals 1 {
    when paused equals 0 {
        pause
        say "Game Paused"
```

```
    } else {
        resume
        say "Resumed"
    }
}
```

56. key / lock / unlock / door

Purpose: Locked doors **Syntax:** `lock door` or `unlock door with key`

```
door_locked is 1

when player_use_door equals 1 {
    when door_locked equals 1 {
        when hasitem "key" equals 1 {
            unlock "door" with "key"
            removeitem "key"
            say "Door unlocked"
        } else {
            say "Door is locked"
        }
    }
}
```

57. trigger / zone

Purpose: Trigger areas **Syntax:** `trigger name at x, y, z radius r`

```
trigger "alarm" at 10, 0, 10 radius 5

when player_in_trigger equals 1 {
    say "Alarm triggered!"
    spawn "enemy" at 15, 0, 15
}
```

58. teleport

Purpose: Instant movement **Syntax:** `teleport to x, y, z`

```
when portal_used equals 1 {
    teleport to 100, 0, 100
    say "Teleported!"
}
```

59. battle / combat

Purpose: Enter combat mode **Syntax:** `battle start` or `combat enable`

```
when enemy_spotted equals 1 {
    battle start
    music "combat_theme"
    enemy_aggressive is 1
}

when all_enemies_dead equals 1 {
    battle end
    music "exploration_theme"
}
```

60. frame / framerate

Purpose: Game loop timing **Syntax:** `frame rate` or `framerate value`

```
framerate 60  # 60 FPS target

# Delta time
delta_time is 1 / 60
movement multiply delta_time
```

PART 5: COMPLETE GAME EXAMPLES

EXAMPLE 1: TEXT ADVENTURE GAME

```
# ===== DUNGEON ADVENTURE =====

say "================================"
say "WELCOME TO THE DARK DUNGEON"
say "================================"
say ""

# Get player info
input "What is your name?" into player_name
say ""
say "Welcome, " player_name "!"
say ""

# Stats
health is 100
```

```
gold is 50
keys is 0

say "You stand at the entrance of a mysterious dungeon."
say "Your health: " health
say "Your gold: " gold
say ""

# Choice 1: Path selection
say "Two paths lie before you:"
say "  1. Left - A dimly lit corridor"
say "  2. Right - A staircase going down"
say ""

input "Which path? (1 or 2)" into path_choice

# Left path
when path_choice equals 1 {
    say ""
    say "You venture down the left corridor..."
    say "The walls are damp and covered in moss."
    say ""

    input "Do you search the walls? (yes/no)" into search

    when search equals "yes" {
        say ""
        say "You find a hidden compartment!"
        gold_found is random 10 30
        gold add gold_found
        say "You found " gold_found " gold!"
        say "Total gold: " gold
    }

    say ""
    say "Ahead, you see a glowing chest!"
    input "Open it? (yes/no)" into open_chest

    when open_chest equals "yes" {
        chance is random 1 100
        when chance greater 50 {
            say ""
            say "It's a treasure chest!"
            treasure is random 50 100
            gold add treasure
            keys add 1
```

```
            say "You found:"
            say "  - " treasure " gold"
            say "  - 1 Key"
        } else {
            say ""
            say "It's a trap!"
            damage is random 20 40
            health subtract damage
            say "You took " damage " damage!"
            say "Health: " health
        }
    }
}

# Right path
when path_choice equals 2 {
    say ""
    say "You descend the dark staircase..."
    say "The air grows cold."
    say ""

    say "At the bottom, a skeleton warrior blocks your path!"
    say ""
    say "COMBAT!"
    say ""

    enemy_health is 50
    your_attack is random 15 25

    repeat 10 times {
        when enemy_health greater 0 {
            # Your turn
            say "You attack for " your_attack " damage!"
            enemy_health subtract your_attack

            when enemy_health greater 0 {
                # Enemy turn
                enemy_attack is random 10 20
                say "Enemy attacks for " enemy_attack " damage!"
                health subtract enemy_attack
                say "Your health: " health
                say "Enemy health: " enemy_health
                say ""

                when health less 1 {
                    say "You have been defeated..."
```

```
                gameover
            }
        } else {
            say "You defeated the skeleton!"
            xp is 100
            loot is 75
            say "  + " xp " XP"
            say "  + " loot " gold"
            gold add loot
        }
    }
}

say ""
say "Beyond the skeleton, you find a legendary sword!"
attack_boost is 10
say "Attack increased by " attack_boost "!"
}

# Final summary
say ""
say "================================"
say "ADVENTURE COMPLETE"
say "================================"
say ""
say player_name "'s Stats:"
say "  Health: " health
say "  Gold: " gold
say "  Keys: " keys
say ""
say "Thanks for playing!"
```

EXAMPLE 2: 2D RPG MAZE GAME

```
# ===== RPG MAZE GAME =====

say "Loading maze..."

# Variables
player_x is 100
player_y is 100
keys_collected is 0
keys_needed is 3
health is 100
```

```
game_active is 1

# Clear and draw
cleargraphics

# Outer walls
drawline from 50, 50 to 550, 50 color "#888888"
drawline from 550, 50 to 550, 550 color "#888888"
drawline from 550, 550 to 50, 550 color "#888888"
drawline from 50, 550 to 50, 50 color "#888888"

# Maze walls
drawline from 150, 100 to 150, 400 color "#666666"
drawline from 250, 100 to 250, 450 color "#666666"
drawline from 350, 150 to 350, 500 color "#666666"
drawline from 450, 100 to 450, 400 color "#666666"

drawline from 100, 150 to 500, 150 color "#666666"
drawline from 100, 250 to 450, 250 color "#666666"
drawline from 150, 350 to 500, 350 color "#666666"
drawline from 100, 450 to 400, 450 color "#666666"

# Goal (exit)
drawrect at 500, 500 size 40, 40 color "#ffd700"
drawtext "EXIT" at 515, 520 color "#000000"

# Keys
drawcircle at 200, 200 radius 10 color "#00ffff"
drawcircle at 400, 300 radius 10 color "#00ffff"
drawcircle at 300, 500 radius 10 color "#00ffff"

# Hazards
drawcircle at 300, 200 radius 15 color "#ff0000"
drawcircle at 450, 450 radius 15 color "#ff0000"

# Player
drawrect at player_x, player_y size 20, 20 color "#00ff00"

# HUD
drawtext "WASD to move" at 60, 30 color "#ffffff"
drawtext "Collect keys!" at 60, 50 color "#00ffff"
drawtext "Health: " health at 60, 70 color "#ff0000"
drawtext "Keys: " keys_collected "/" keys_needed at 60, 90 color "#ffff00"

say ""
say "Maze ready!"
```

```
say "Commands:"
say "  player_x add 20    (move right)"
say "  player_x subtract 20 (move left)"
say "  player_y add 20    (move down)"
say "  player_y subtract 20 (move up)"
say ""
say "Then re-run this script!"
```

EXAMPLE 3: 3D FPS GAME

```
# ===== 3D FPS GAME =====

say "Loading FPS game..."

# Player stats
health is 100
ammo is 30
score is 0
kills is 0

# Enemies
enemy_count is 5

# Create arena
create3d plane at 0, -1, 0 size 20
color3d last3d to "#404040"

# Walls
create3d cube at 10, 0, 0 size 1, 2, 20
color3d last3d to "#808080"

create3d cube at -10, 0, 0 size 1, 2, 20
color3d last3d to "#808080"

create3d cube at 0, 0, 10 size 20, 2, 1
color3d last3d to "#808080"

create3d cube at 0, 0, -10 size 20, 2, 1
color3d last3d to "#808080"

# Spawn enemies
create3d cube at 5, 0, 8 size 1
color3d last3d to "#ff0000"
enemy "enemy1" at 5, 0, 8 health 50
```

```
create3d cube at -5, 0, 8 size 1
color3d last3d to "#ff0000"
enemy "enemy2" at -5, 0, 8 health 50

create3d cube at 7, 0, 3 size 1
color3d last3d to "#ff0000"
enemy "enemy3" at 7, 0, 3 health 50

create3d cube at -7, 0, 3 size 1
color3d last3d to "#ff0000"
enemy "enemy4" at -7, 0, 3 health 50

create3d cube at 0, 0, 5 size 1
color3d last3d to "#ff0000"
enemy "enemy5" at 0, 0, 5 health 50

# Lighting
pointlight at 0, 5, 0 color "#ffffff" intensity 1.0

# Camera
camera at 0, 1.6, -5

say ""
say "FPS Game Ready!"
say ""
say "CONTROLS:"
say "  Enter Shooter mode in 3D Viewport"
say "  WASD - Move"
say "  Mouse - Look"
say "  Left Click - Shoot"
say ""
say "OBJECTIVE:"
say "  Eliminate all enemies!"
say ""
say "STATS:"
say "  Health: " health
say "  Ammo: " ammo
say "  Kills: " kills "/" enemy_count
```

EXAMPLE 4: SIMPLE PHYSICS GAME

```
# ===== PHYSICS GAME =====
```

```
say "Loading physics game..."

cleargraphics

# Ball
ball_x is 300
ball_y is 100
ball_vx is 5
ball_vy is 0
ball_size is 20

gravity is 0.5

# Ground
ground_y is 500

# Draw game
drawcircle at ball_x, ball_y radius ball_size color "#00ff00"
drawline from 0, ground_y to 600, ground_y color "#ffffff"

# Text
drawtext "Press SPACE to boost up" at 200, 50 color "#ffffff"
drawtext "Don't hit the ground!" at 200, 70 color "#ff0000"

say ""
say "Physics game ready!"
say ""
say "Simulation:"
say "  Ball starts at top"
say "  Gravity pulls down"
say "  Bounces on ground"
say ""
say "Commands to run simulation:"
say "  ball_vy add gravity"
say "  ball_y add ball_vy"
say "  when ball_y greater ground_y {"
say "    ball_vy multiply -0.8"
say "  }"
```

TO BE CONTINUED IN PART 5… (Advanced Examples, Tips & Tricks, Troubleshooting, and Reference Tables)