# Mission to Mars - Slide Text Content (8 Lessons × 7 Slides)

## LESSON 1: OBJECT POSITIONING AND SCALING (7 SLIDES)

### SLIDE 1: LESSON TITLE

**Mission to Mars - Lesson 1: Object Positioning and Scaling**

**Learning Objectives:**

* Position objects in 3D space using X, Y, Z coordinates
* Scale objects to realistic proportions
* Understand transform controls for object manipulation
* Create spatial relationships between celestial objects

**Key Computing Concepts:** 3D Positioning, Scaling, Transform Controls, Coordinate Systems

### SLIDE 2: YOUR MISSION

**Your Mission Today**

Learn to place and size objects in 3D space using **transform controls**. You'll master positioning objects using X, Y, Z coordinates and scaling them to realistic sizes. By the end of this lesson, you should be able to create a static solar system with correctly positioned and scaled planets.

**Success Looks Like:** Objects placed at correct positions with realistic size relationships (large sun, medium planets, small moons).

### SLIDE 3: CONCEPT EXPLANATION

**Understanding 3D Space and Transform Controls**

**Coordinate System:**

* X coordinate: left (-) and right (+) positioning
* Y coordinate: down (-) and up (+) positioning
* Z coordinate: backward (-) and forward (+) positioning

**Scaling Concepts:**

* Scale value 1.0 = normal size
* Scale above 1.0 = larger objects
* Scale below 1.0 = smaller objects

**Real-World Connection:** Astronomers use coordinate systems to map celestial objects in space

### SLIDE 4: CORE ACTIVITY

**Step-by-Step Object Positioning**

**Step 1: Understanding Coordinates**

* Open transform menu for each object
* Practice moving objects using X, Y, Z values
* Observe how coordinate changes affect position

**Step 2: Solar System Setup**

* Place sun at center (0, 0, 0)
* Position Earth at (5, 0, 0)
* Position Moon at (6, 0, 0)

**Step 3: Realistic Scaling**

* Sun: scale 3.0 (largest object)
* Earth: scale 1.0 (reference size)
* Moon: scale 0.3 (smallest object)

### SLIDE 5: MINI PLENARY

**Quick Assessment Questions**

1. Which coordinate moves objects left and right? (X coordinate)
2. What scale value makes objects twice as large? (2.0)
3. Where should you position the sun in your solar system? (Center at 0, 0, 0)
4. Why is scaling important for realistic solar systems? (Shows size relationships)

**Check Your Understanding:**

* Can you position an object exactly where you want it?
* How do you make realistic size differences between objects?

### SLIDE 6: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 7: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Show your positioned solar system with clear size relationships
* Explain your coordinate choices for object placement
* Demonstrate understanding of scale values

**Reflection Questions:**

* What was most challenging about 3D positioning?
* How did you decide on realistic scale relationships?
* Which coordinate was trickiest to understand?

**Next Lesson Preview:** Making your positioned objects move along simple paths

## LESSON 2: SIMPLE ANIMATION PATHS (7 SLIDES)

### SLIDE 1: LESSON TITLE

**Mission to Mars - Lesson 2: Simple Animation Paths**

**Learning Objectives:**

* Create simple path-based movement for objects
* Use move on path blocks for smooth animation
* Control animation timing with duration settings
* Understand the relationship between paths and movement

**Key Computing Concepts:** Paths, Move on Path, Duration, Animation Timing

### SLIDE 2: YOUR MISSION

**Your Mission Today**

Make your positioned objects move along simple paths using **move on path** blocks. You'll create smooth orbital motion for planets around the sun. By the end of this lesson, your solar system should have planets that move in continuous circles.

**Success Looks Like:** Planets moving smoothly in circular paths without stopping or overlapping.

### SLIDE 3: CONCEPT EXPLANATION

**Understanding Paths and Movement**

**Path Concepts:**

* Paths are invisible routes objects follow
* Circular paths create orbital motion
* Path size determines orbit distance

**Movement Controls:**

* Move on path block controls object movement
* Duration setting controls movement speed
* Shorter duration = faster movement

**Real-World Connection:** Planets follow elliptical paths around the sun due to gravitational forces

### SLIDE 4: CORE ACTIVITY

**Step-by-Step Animation Creation**

**Step 1: Path Creation**

* Create circular path around sun for Earth
* Size path appropriately for orbital distance
* Test path visibility and shape

**Step 2: Movement Implementation**

move Earth on path EarthOrbit duration 10s

**Step 3: Testing and Adjustment**

* Run animation once to test
* Adjust duration for realistic orbital speed
* Verify smooth movement along entire path

### SLIDE 5: MINI PLENARY

**Quick Assessment Questions**

1. What determines how fast an object moves on a path? (Duration setting)
2. What shape path creates orbital motion? (Circular)
3. How do you make an object move slower? (Increase duration number)
4. What happens if your path is too small? (Object moves too close to sun)

**Check Your Understanding:**

* Can you create smooth orbital motion?
* How does duration affect movement speed?

### SLIDE 6: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 7: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Demonstrate smooth planet movement along orbital path
* Show understanding of duration and speed relationship
* Explain path size choices for realistic orbits

**Reflection Questions:**

* What made creating smooth movement challenging?
* How did you choose appropriate path sizes?
* What would you change about your orbital speeds?

**Next Lesson Preview:** Making your animations run continuously forever

## LESSON 3: CONTINUOUS LOOPS (7 SLIDES)

### SLIDE 1: LESSON TITLE

**Mission to Mars - Lesson 3: Continuous Loops**

**Learning Objectives:**

* Implement forever loops for continuous animation
* Understand the difference between single and repeated actions
* Create ongoing orbital motion that doesn't stop
* Combine loops with path-based movement

**Key Computing Concepts:** Forever Loops, Continuous Animation, Sequence, Repetition

### SLIDE 2: YOUR MISSION

**Your Mission Today**

Make your animations run continuously using **forever loops**. You'll transform single movements into ongoing orbital motion. By the end of this lesson, your planets should orbit continuously without stopping.

**Success Looks Like:** Planets that orbit forever without manual restart, creating realistic continuous motion.

### SLIDE 3: CONCEPT EXPLANATION

**Understanding Continuous Loops**

**Forever Loop Concepts:**

* Forever blocks repeat actions endlessly
* Create continuous animation cycles
* Simulate real planetary motion

**Loop Structure:**

* Forever block contains other commands
* Commands inside repeat automatically
* Loop continues until manually stopped

**Real-World Connection:** Real planets orbit continuously due to gravitational forces and momentum

### SLIDE 4: CORE ACTIVITY

**Step-by-Step Loop Implementation**

**Step 1: Adding Forever Blocks**

forever

move Earth on path EarthOrbit duration 10s

**Step 2: Testing Continuous Motion**

* Start animation and observe
* Verify smooth continuous movement
* Check for any stopping or stuttering

**Step 3: Multiple Object Loops**

* Add forever loops for Moon around Earth
* Test coordination between multiple loops
* Ensure smooth simultaneous motion

### SLIDE 5: MINI PLENARY

**Quick Assessment Questions**

1. What block makes animations repeat continuously? (Forever block)
2. What goes inside a forever loop? (The actions you want to repeat)
3. How do you stop a forever loop? (Stop the program)
4. Why do we use forever loops for planetary motion? (Planets orbit continuously)

**Check Your Understanding:**

* Can you create motion that never stops?
* What's the difference between single movement and continuous loops?

### SLIDE 6: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 7: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Show planets orbiting continuously without stopping
* Demonstrate understanding of forever loop structure
* Explain why continuous motion is important for realism

**Reflection Questions:**

* How does continuous motion change your solar system?
* What problems did you solve with forever loops?
* Why is continuous animation more realistic?

**Next Lesson Preview:** Coordinating multiple objects moving simultaneously

## LESSON 4: MULTIPLE OBJECT ANIMATION (7 SLIDES)

### SLIDE 1: LESSON TITLE

**Mission to Mars - Lesson 4: Multiple Object Animation**

**Learning Objectives:**

* Coordinate multiple animations running simultaneously
* Use parallel processing for complex motion systems
* Create realistic speed relationships between objects
* Manage timing for multiple moving elements

**Key Computing Concepts:** Parallel Processing, Multiple Animations, Timing Coordination, Speed Relationships

### SLIDE 2: YOUR MISSION

**Your Mission Today**

Coordinate multiple objects moving simultaneously using **parallel processing**. You'll create a complete solar system where several planets orbit at the same time. By the end of this lesson, your solar system should show realistic multi-planet motion.

**Success Looks Like:** Multiple planets orbiting simultaneously with different speeds, creating a dynamic solar system.

### SLIDE 3: CONCEPT EXPLANATION

**Understanding Multiple Animations**

**Parallel Processing:**

* Multiple forever loops running at the same time
* Each object has its own animation instructions
* All animations start together

**Speed Relationships:**

* Inner planets orbit faster (shorter duration)
* Outer planets orbit slower (longer duration)
* Creates realistic astronomical motion

**Real-World Connection:** In our solar system, Mercury completes an orbit in 88 days while Neptune takes 165 years

### SLIDE 4: CORE ACTIVITY

**Step-by-Step Multi-Object Animation**

**Step 1: Parallel Animation Setup**

run parallel

forever

move Mercury on path MercuryOrbit duration 5s

run parallel

forever

move Earth on path EarthOrbit duration 10s

**Step 2: Speed Relationships**

* Inner planets: shorter durations (faster orbits)
* Outer planets: longer durations (slower orbits)
* Create logical progression of speeds

**Step 3: System Testing**

* Run all animations together
* Verify realistic speed relationships
* Adjust timing for best visual effect

### SLIDE 5: MINI PLENARY

**Quick Assessment Questions**

1. How do you make multiple objects move at the same time? (Parallel processing)
2. Which planets should move faster? (Inner planets, closer to sun)
3. What happens if all planets have the same duration? (Unrealistic - all same speed)
4. How do you create realistic speed differences? (Different duration values)

**Check Your Understanding:**

* Can you coordinate multiple moving objects?
* How do you create realistic planetary speed relationships?

### SLIDE 6: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 7: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Show multiple planets orbiting with realistic speed differences
* Demonstrate understanding of parallel processing
* Explain your speed relationship choices

**Reflection Questions:**

* What was challenging about coordinating multiple animations?
* How did you decide on realistic speed relationships?
* What makes your solar system look more realistic?

**Next Lesson Preview:** Controlling object movement with button clicks

## LESSON 5: SIMPLE BUTTON CONTROLS (7 SLIDES)

### SLIDE 1: LESSON TITLE

**Mission to Mars - Lesson 5: Simple Button Controls**

**Learning Objectives:**

* Implement event-driven programming with button controls
* Use push blocks for directional object movement
* Create responsive user input systems
* Control velocity and stopping mechanisms

**Key Computing Concepts:** Event-Driven Programming, Button Controls, Velocity, User Input

### SLIDE 2: YOUR MISSION

**Your Mission Today**

Control object movement using **button clicks** and **push blocks**. You'll create an astronaut that responds to directional buttons. By the end of this lesson, you should have precise control over astronaut movement in space.

**Success Looks Like:** An astronaut that moves reliably in response to button presses with appropriate stopping control.

### SLIDE 3: CONCEPT EXPLANATION

**Understanding Button Controls and Movement**

**Event-Driven Programming:**

* Button clicks trigger specific actions
* Each button controls one direction
* User input creates interactive responses

**Velocity Control:**

* Push blocks create movement with force and direction
* Velocity value controls movement speed
* Stop command sets velocity to zero

**Real-World Connection:** Astronauts use thruster controls for precise movement in zero gravity environments

### SLIDE 4: CORE ACTIVITY

**Step-by-Step Button Control Creation**

**Step 1: Single Direction Control**

when Up is clicked

push Astronaut up with velocity 1

**Step 2: Adding Stop Control**

when Stop is clicked

push Astronaut stop with velocity 0

**Step 3: Multiple Directions**

* Add buttons for down, left, right
* Test each direction individually
* Verify all controls respond correctly

### SLIDE 5: MINI PLENARY

**Quick Assessment Questions**

1. What triggers an action in event-driven programming? (Button clicks or user input)
2. How do you stop object movement? (Push with velocity 0)
3. What determines movement speed? (Velocity value)
4. Why is a stop button important? (Prevents objects floating away forever)

**Check Your Understanding:**

* Can you control object movement precisely?
* How do button clicks create responsive interactions?

### SLIDE 6: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 7: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Demonstrate precise astronaut control in all directions
* Show responsive button-based movement system
* Explain velocity and stopping mechanisms

**Reflection Questions:**

* How does button control differ from automatic animation?
* What makes user input systems responsive?
* Which direction controls were most challenging to set up?

**Next Lesson Preview:** Creating smart objects that remember states using true/false logic

## LESSON 6: GRAVITY TOGGLE BOX (7 SLIDES)

### SLIDE 1: LESSON TITLE

**Mission to Mars - Lesson 6: Gravity Toggle Box**

**Learning Objectives:**

* Implement boolean variables for state tracking
* Use if/else logic for conditional responses
* Control gravity physics with toggle systems
* Combine visual feedback with physics changes

**Key Computing Concepts:** Boolean Variables, If/Else Logic, Gravity Control, State Management

### SLIDE 2: YOUR MISSION

**Your Mission Today**

Create a box that controls gravity using **boolean variables** and **if/else logic**. When clicked, the box will float up as gravity turns off, then fall back down when clicked again as gravity turns on. You'll combine state management with physics control.

**Success Looks Like:** A clickable box that changes color AND controls gravity - floating up when gravity is off, falling down when gravity is on.

### SLIDE 3: CONCEPT EXPLANATION

**Understanding Boolean Logic and Physics Control**

**Boolean Variables:**

* True/false states for tracking conditions
* Variables that remember on/off states
* Essential for toggle behaviors

**If/Else Logic:**

* Decision making based on current state
* Different actions for true vs false conditions
* Creates intelligent responsive systems

**Real-World Connection:** Spacecraft systems use boolean logic to control life support, gravity simulation, and safety systems

### SLIDE 4: CORE ACTIVITY

**Step-by-Step Gravity Toggle Creation**

**Step 1: Initial Setup**

when play clicked

set variable box to false

set gravity pull to 10

set color of Wooden box to blue

**Step 2: Toggle Logic with Gravity Control**

when Wooden box is clicked

if box = false

set variable box to true

set gravity pull to 0

set color of Wooden box to red

else

set variable box to false

set gravity pull to 10

set color of Wooden box to blue

**Step 3: Testing Physics Response**

* Click box and observe floating behavior
* Click again and observe falling behavior
* Verify color matches gravity state

### SLIDE 5: MINI PLENARY

**Quick Assessment Questions**

1. What are the two possible values for boolean variables? (True and false)
2. How does if/else logic make decisions? (Checks condition, does different actions)
3. What visual feedback shows the gravity state? (Color change - blue/red)
4. Why does the box float when gravity is off? (No downward force)

**Check Your Understanding:**

* Can you create toggle behavior with boolean logic?
* How do visual cues help users understand system states?

### SLIDE 6: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 7: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Demonstrate gravity toggle with visual feedback
* Show box floating and falling based on state
* Explain how boolean logic controls physics

**Reflection Questions:**

* How does boolean logic make objects "smart"?
* What makes toggle systems useful for control?
* How do visual cues improve user experience?

**Next Lesson Preview:** Building a Mars control room to manage multiple objects

## LESSON 7: MARS CONTROL ROOM WITH MULTIPLE OBJECTS (7 SLIDES)

### SLIDE 1: LESSON TITLE

**Mission to Mars - Lesson 7: Mars Control Room with Multiple Objects**

**Learning Objectives:**

* Organize multiple objects using lists
* Create functions for reusable code blocks
* Control gravity for multiple objects simultaneously
* Build efficient control systems for complex environments

**Key Computing Concepts:** Lists, Functions, Code Organization, Multiple Object Control

### SLIDE 2: YOUR MISSION

**Your Mission Today**

Build a Mars control room using **lists** and **functions** to control multiple objects simultaneously. You'll create a gravity control system where pressing one button turns gravity off for all objects (making them float), and another button turns gravity back on (making them fall).

**Success Looks Like:** A control room where multiple objects respond to gravity changes together, demonstrating how functions and lists organize complex behaviors.

### SLIDE 3: CONCEPT EXPLANATION

**Understanding Lists and Functions for Organization**

**Lists:**

* Collections of objects for group management
* Efficient way to control multiple items
* Add objects once, affect them all together

**Functions:**

* Reusable code blocks for organizing actions
* Reduce repetition and improve organization
* Can be called multiple times

**Real-World Connection:** Mars mission control centers use automated systems to manage multiple spacecraft systems simultaneously

### SLIDE 4: CORE ACTIVITY

**Step-by-Step Control Room Creation**

**Step 1: Creating Object Lists**

when play clicked

create empty list gravityItems

add Crate to gravityItems

add Tool to gravityItems

add Equipment to gravityItems

**Step 2: Gravity Control Functions**

define function turnGravityOff()

set gravity pull to 0

for each element in gravityItems

set color of element to blue

define function turnGravityOn()

set gravity pull to 10

for each element in gravityItems

set color of element to red

**Step 3: Button Control System**

when GravityOff is clicked

call function turnGravityOff()

when GravityOn is clicked

call function turnGravityOn()

### SLIDE 5: MINI PLENARY

**Quick Assessment Questions**

1. How do you add objects to a list? (Add [object] to [list name])
2. What makes functions reusable? (Can be called multiple times)
3. How do you affect all objects in a list? (For each element loop)
4. Why organize code with functions? (Reduces repetition, improves clarity)

**Check Your Understanding:**

* Can you control multiple objects efficiently?
* How do lists make managing groups easier?

### SLIDE 6: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 7: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Demonstrate gravity control affecting multiple objects
* Show organized code using functions and lists
* Explain efficiency benefits of your control system

**Reflection Questions:**

* How do functions make your code better organized?
* What advantages do lists provide for multiple objects?
* How does your control room simulate real Mars systems?

**Next Lesson Preview:** Integrating all skills to create a complete Mars base

## LESSON 8: MARS BASE INTEGRATION (7 SLIDES)

### SLIDE 1: LESSON TITLE

**Mission to Mars - Lesson 8: Mars Base Integration**

**Learning Objectives:**

* Integrate all learned programming concepts
* Create complex interactive environments
* Apply positioning, animation, controls, and logic together
* Design creative solutions for Mars base systems

**Key Computing Concepts:** Integration, Project Planning, Complex Systems, Creative Application

### SLIDE 2: YOUR MISSION

**Your Mission Today**

Combine all learned skills to create an interactive **Mars base** with **camera tours** and **simple interactions**. You'll integrate positioning, animation, controls, and functions into one complete project. By the end of this lesson, you should have a working Mars base that showcases all your coding skills.

**Success Looks Like:** A complete Mars base demonstrating integration of all previous concepts with guided tours and interactive elements.

### SLIDE 3: CONCEPT EXPLANATION

**Understanding System Integration**

**Integration Concepts:**

* Combining multiple programming skills
* Using positioning, animation, controls, and logic together
* Creating complex interactive environments

**Project Planning:**

* Design before coding
* Plan how different systems work together
* Test components before combining

**Real-World Connection:** Real Mars bases will require integrated systems for life support, research, communication, and exploration

### SLIDE 4: CORE ACTIVITY

**Step-by-Step Mars Base Creation**

**Step 1: Environment Setup**

* Use Week 1 skills: position and scale base structures
* Apply realistic Mars environment design
* Plan base layout and key areas

**Step 2: Animation Integration**

* Use Week 2-4 skills: create moving elements
* Add simple camera tour path
* Include rotating equipment or moving vehicles

**Step 3: Interactive Elements**

* Use Week 5-6 skills: add button controls and toggles
* Include gravity control systems
* Add responsive user interactions

**Step 4: Function Integration**

* Use Week 7 skills: organize complex behaviors
* Apply functions to manage multiple interactive elements
* Test all systems working together

### SLIDE 5: MINI PLENARY

**Quick Assessment Questions**

1. Which skills from previous weeks are you combining? (All: positioning, animation, controls, logic, functions)
2. How do you plan complex projects? (Design first, test components, integrate gradually)
3. What makes a Mars base realistic? (Multiple functional areas, interactive systems)
4. Why is integration more challenging than individual skills? (Multiple systems must work together)

**Check Your Understanding:**

* Can you combine multiple programming concepts effectively?
* How do you troubleshoot when complex systems don't work?

### SLIDE 6: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 7: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Present your complete Mars base with guided tour
* Demonstrate integration of all learned concepts
* Explain design decisions and problem-solving approaches

**Final Unit Reflection Questions:**

* Which programming concept was most transformative for your projects?
* How did your problem-solving skills develop throughout the unit?
* What would you create next using these coding skills?
* How do these skills connect to real Mars exploration technology?

**Computing Vocabulary Mastery:** 3D Positioning, Animation Paths, Forever Loops, Parallel Processing, Event-Driven Programming, Boolean Variables, If/Else Logic, Lists, Functions, System Integration

## LESSON 1: OBJECT POSITIONING AND SCALING (7 SLIDES)

### SLIDE 1: LESSON TITLE

**Mission to Mars - Lesson 1: Object Positioning and Scaling**

**Learning Objectives:**

* Position objects in 3D space using X, Y, Z coordinates
* Scale objects to realistic proportions
* Understand transform controls for object manipulation
* Create spatial relationships between celestial objects

**Key Computing Concepts:** 3D Positioning, Scaling, Transform Controls, Coordinate Systems

### SLIDE 2: YOUR MISSION

**Your Mission Today**

Learn to place and size objects in 3D space using **transform controls**. You'll master positioning objects using X, Y, Z coordinates and scaling them to realistic sizes. By the end of this lesson, you should be able to create a static solar system with correctly positioned and scaled planets.

**Success Looks Like:** Objects placed at correct positions with realistic size relationships (large sun, medium planets, small moons).

### SLIDE 3: CONCEPT EXPLANATION

**Understanding 3D Space and Transform Controls**

**Coordinate System:**

* X coordinate: left (-) and right (+) positioning
* Y coordinate: down (-) and up (+) positioning
* Z coordinate: backward (-) and forward (+) positioning

**Scaling Concepts:**

* Scale value 1.0 = normal size
* Scale above 1.0 = larger objects
* Scale below 1.0 = smaller objects

**Real-World Connection:** Astronomers use coordinate systems to map celestial objects in space

### SLIDE 4: CORE ACTIVITY

**Step-by-Step Object Positioning**

**Step 1: Understanding Coordinates**

* Open transform menu for each object
* Practice moving objects using X, Y, Z values
* Observe how coordinate changes affect position

**Step 2: Solar System Setup**

* Place sun at center (0, 0, 0)
* Position Earth at (5, 0, 0)
* Position Moon at (6, 0, 0)

**Step 3: Realistic Scaling**

* Sun: scale 3.0 (largest object)
* Earth: scale 1.0 (reference size)
* Moon: scale 0.3 (smallest object)

### SLIDE 5: MINI PLENARY

**Quick Assessment Questions**

1. Which coordinate moves objects left and right? (X coordinate)
2. What scale value makes objects twice as large? (2.0)
3. Where should you position the sun in your solar system? (Center at 0, 0, 0)
4. Why is scaling important for realistic solar systems? (Shows size relationships)

**Check Your Understanding:**

* Can you position an object exactly where you want it?
* How do you make realistic size differences between objects?

### SLIDE 6: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 7: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Show your positioned solar system with clear size relationships
* Explain your coordinate choices for object placement
* Demonstrate understanding of scale values

**Reflection Questions:**

* What was most challenging about 3D positioning?
* How did you decide on realistic scale relationships?
* Which coordinate was trickiest to understand?

**Next Lesson Preview:** Making your positioned objects move along simple paths

## LESSON 2: SIMPLE ANIMATION PATHS (7 SLIDES)

### SLIDE 8: LESSON TITLE

**Mission to Mars - Lesson 2: Simple Animation Paths**

**Learning Objectives:**

* Create simple path-based movement for objects
* Use move on path blocks for smooth animation
* Control animation timing with duration settings
* Understand the relationship between paths and movement

**Key Computing Concepts:** Paths, Move on Path, Duration, Animation Timing

### SLIDE 9: YOUR MISSION

**Your Mission Today**

Make your positioned objects move along simple paths using **move on path** blocks. You'll create smooth orbital motion for planets around the sun. By the end of this lesson, your solar system should have planets that move in continuous circles.

**Success Looks Like:** Planets moving smoothly in circular paths without stopping or overlapping.

### SLIDE 10: CONCEPT EXPLANATION

**Understanding Paths and Movement**

**Path Concepts:**

* Paths are invisible routes objects follow
* Circular paths create orbital motion
* Path size determines orbit distance

**Movement Controls:**

* Move on path block controls object movement
* Duration setting controls movement speed
* Shorter duration = faster movement

**Real-World Connection:** Planets follow elliptical paths around the sun due to gravitational forces

### SLIDE 11: CORE ACTIVITY

**Step-by-Step Animation Creation**

**Step 1: Path Creation**

* Create circular path around sun for Earth
* Size path appropriately for orbital distance
* Test path visibility and shape

**Step 2: Movement Implementation**

move Earth on path EarthOrbit duration 10s

**Step 3: Testing and Adjustment**

* Run animation once to test
* Adjust duration for realistic orbital speed
* Verify smooth movement along entire path

### SLIDE 12: MINI PLENARY

**Quick Assessment Questions**

1. What determines how fast an object moves on a path? (Duration setting)
2. What shape path creates orbital motion? (Circular)
3. How do you make an object move slower? (Increase duration number)
4. What happens if your path is too small? (Object moves too close to sun)

**Check Your Understanding:**

* Can you create smooth orbital motion?
* How does duration affect movement speed?

### SLIDE 13: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 14: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Demonstrate smooth planet movement along orbital path
* Show understanding of duration and speed relationship
* Explain path size choices for realistic orbits

**Reflection Questions:**

* What made creating smooth movement challenging?
* How did you choose appropriate path sizes?
* What would you change about your orbital speeds?

**Next Lesson Preview:** Making your animations run continuously forever

## LESSON 3: CONTINUOUS LOOPS (7 SLIDES)

### SLIDE 15: LESSON TITLE

**Mission to Mars - Lesson 3: Continuous Loops**

**Learning Objectives:**

* Implement forever loops for continuous animation
* Understand the difference between single and repeated actions
* Create ongoing orbital motion that doesn't stop
* Combine loops with path-based movement

**Key Computing Concepts:** Forever Loops, Continuous Animation, Sequence, Repetition

### SLIDE 16: YOUR MISSION

**Your Mission Today**

Make your animations run continuously using **forever loops**. You'll transform single movements into ongoing orbital motion. By the end of this lesson, your planets should orbit continuously without stopping.

**Success Looks Like:** Planets that orbit forever without manual restart, creating realistic continuous motion.

### SLIDE 17: CONCEPT EXPLANATION

**Understanding Continuous Loops**

**Forever Loop Concepts:**

* Forever blocks repeat actions endlessly
* Create continuous animation cycles
* Simulate real planetary motion

**Loop Structure:**

* Forever block contains other commands
* Commands inside repeat automatically
* Loop continues until manually stopped

**Real-World Connection:** Real planets orbit continuously due to gravitational forces and momentum

### SLIDE 18: CORE ACTIVITY

**Step-by-Step Loop Implementation**

**Step 1: Adding Forever Blocks**

forever

move Earth on path EarthOrbit duration 10s

**Step 2: Testing Continuous Motion**

* Start animation and observe
* Verify smooth continuous movement
* Check for any stopping or stuttering

**Step 3: Multiple Object Loops**

* Add forever loops for Moon around Earth
* Test coordination between multiple loops
* Ensure smooth simultaneous motion

### SLIDE 19: MINI PLENARY

**Quick Assessment Questions**

1. What block makes animations repeat continuously? (Forever block)
2. What goes inside a forever loop? (The actions you want to repeat)
3. How do you stop a forever loop? (Stop the program)
4. Why do we use forever loops for planetary motion? (Planets orbit continuously)

**Check Your Understanding:**

* Can you create motion that never stops?
* What's the difference between single movement and continuous loops?

### SLIDE 20: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 21: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Show planets orbiting continuously without stopping
* Demonstrate understanding of forever loop structure
* Explain why continuous motion is important for realism

**Reflection Questions:**

* How does continuous motion change your solar system?
* What problems did you solve with forever loops?
* Why is continuous animation more realistic?

**Next Lesson Preview:** Coordinating multiple objects moving simultaneously

## LESSON 4: MULTIPLE OBJECT ANIMATION (7 SLIDES)

### SLIDE 22: LESSON TITLE

**Mission to Mars - Lesson 4: Multiple Object Animation**

**Learning Objectives:**

* Coordinate multiple animations running simultaneously
* Use parallel processing for complex motion systems
* Create realistic speed relationships between objects
* Manage timing for multiple moving elements

**Key Computing Concepts:** Parallel Processing, Multiple Animations, Timing Coordination, Speed Relationships

### SLIDE 23: YOUR MISSION

**Your Mission Today**

Coordinate multiple objects moving simultaneously using **parallel processing**. You'll create a complete solar system where several planets orbit at the same time. By the end of this lesson, your solar system should show realistic multi-planet motion.

**Success Looks Like:** Multiple planets orbiting simultaneously with different speeds, creating a dynamic solar system.

### SLIDE 24: CONCEPT EXPLANATION

**Understanding Multiple Animations**

**Parallel Processing:**

* Multiple forever loops running at the same time
* Each object has its own animation instructions
* All animations start together

**Speed Relationships:**

* Inner planets orbit faster (shorter duration)
* Outer planets orbit slower (longer duration)
* Creates realistic astronomical motion

**Real-World Connection:** In our solar system, Mercury completes an orbit in 88 days while Neptune takes 165 years

### SLIDE 25: CORE ACTIVITY

**Step-by-Step Multi-Object Animation**

**Step 1: Parallel Animation Setup**

run parallel

forever

move Mercury on path MercuryOrbit duration 5s

run parallel

forever

move Earth on path EarthOrbit duration 10s

**Step 2: Speed Relationships**

* Inner planets: shorter durations (faster orbits)
* Outer planets: longer durations (slower orbits)
* Create logical progression of speeds

**Step 3: System Testing**

* Run all animations together
* Verify realistic speed relationships
* Adjust timing for best visual effect

### SLIDE 26: MINI PLENARY

**Quick Assessment Questions**

1. How do you make multiple objects move at the same time? (Parallel processing)
2. Which planets should move faster? (Inner planets, closer to sun)
3. What happens if all planets have the same duration? (Unrealistic - all same speed)
4. How do you create realistic speed differences? (Different duration values)

**Check Your Understanding:**

* Can you coordinate multiple moving objects?
* How do you create realistic planetary speed relationships?

### SLIDE 27: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 28: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Show multiple planets orbiting with realistic speed differences
* Demonstrate understanding of parallel processing
* Explain your speed relationship choices

**Reflection Questions:**

* What was challenging about coordinating multiple animations?
* How did you decide on realistic speed relationships?
* What makes your solar system look more realistic?

**Next Lesson Preview:** Controlling object movement with button clicks

## LESSON 5: SIMPLE BUTTON CONTROLS (7 SLIDES)

### SLIDE 29: LESSON TITLE

**Mission to Mars - Lesson 5: Simple Button Controls**

**Learning Objectives:**

* Implement event-driven programming with button controls
* Use push blocks for directional object movement
* Create responsive user input systems
* Control velocity and stopping mechanisms

**Key Computing Concepts:** Event-Driven Programming, Button Controls, Velocity, User Input

### SLIDE 30: YOUR MISSION

**Your Mission Today**

Control object movement using **button clicks** and **push blocks**. You'll create an astronaut that responds to directional buttons. By the end of this lesson, you should have precise control over astronaut movement in space.

**Success Looks Like:** An astronaut that moves reliably in response to button presses with appropriate stopping control.

### SLIDE 31: CONCEPT EXPLANATION

**Understanding Button Controls and Movement**

**Event-Driven Programming:**

* Button clicks trigger specific actions
* Each button controls one direction
* User input creates interactive responses

**Velocity Control:**

* Push blocks create movement with force and direction
* Velocity value controls movement speed
* Stop command sets velocity to zero

**Real-World Connection:** Astronauts use thruster controls for precise movement in zero gravity environments

### SLIDE 32: CORE ACTIVITY

**Step-by-Step Button Control Creation**

**Step 1: Single Direction Control**

when Up is clicked

push Astronaut up with velocity 1

**Step 2: Adding Stop Control**

when Stop is clicked

push Astronaut stop with velocity 0

**Step 3: Multiple Directions**

* Add buttons for down, left, right
* Test each direction individually
* Verify all controls respond correctly

### SLIDE 33: MINI PLENARY

**Quick Assessment Questions**

1. What triggers an action in event-driven programming? (Button clicks or user input)
2. How do you stop object movement? (Push with velocity 0)
3. What determines movement speed? (Velocity value)
4. Why is a stop button important? (Prevents objects floating away forever)

**Check Your Understanding:**

* Can you control object movement precisely?
* How do button clicks create responsive interactions?

### SLIDE 34: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 35: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Demonstrate precise astronaut control in all directions
* Show responsive button-based movement system
* Explain velocity and stopping mechanisms

**Reflection Questions:**

* How does button control differ from automatic animation?
* What makes user input systems responsive?
* Which direction controls were most challenging to set up?

**Next Lesson Preview:** Creating smart objects that remember states using true/false logic

## LESSON 6: GRAVITY TOGGLE BOX (7 SLIDES)

### SLIDE 36: LESSON TITLE

**Mission to Mars - Lesson 6: Gravity Toggle Box**

**Learning Objectives:**

* Implement boolean variables for state tracking
* Use if/else logic for conditional responses
* Control gravity physics with toggle systems
* Combine visual feedback with physics changes

**Key Computing Concepts:** Boolean Variables, If/Else Logic, Gravity Control, State Management

### SLIDE 37: YOUR MISSION

**Your Mission Today**

Create a box that controls gravity using **boolean variables** and **if/else logic**. When clicked, the box will float up as gravity turns off, then fall back down when clicked again as gravity turns on. You'll combine state management with physics control.

**Success Looks Like:** A clickable box that changes color AND controls gravity - floating up when gravity is off, falling down when gravity is on.

### SLIDE 38: CONCEPT EXPLANATION

**Understanding Boolean Logic and Physics Control**

**Boolean Variables:**

* True/false states for tracking conditions
* Variables that remember on/off states
* Essential for toggle behaviors

**If/Else Logic:**

* Decision making based on current state
* Different actions for true vs false conditions
* Creates intelligent responsive systems

**Real-World Connection:** Spacecraft systems use boolean logic to control life support, gravity simulation, and safety systems

### SLIDE 39: CORE ACTIVITY

**Step-by-Step Gravity Toggle Creation**

**Step 1: Initial Setup**

when play clicked

set variable box to false

set gravity pull to 10

set color of Wooden box to blue

**Step 2: Toggle Logic with Gravity Control**

when Wooden box is clicked

if box = false

set variable box to true

set gravity pull to 0

set color of Wooden box to red

else

set variable box to false

set gravity pull to 10

set color of Wooden box to blue

**Step 3: Testing Physics Response**

* Click box and observe floating behavior
* Click again and observe falling behavior
* Verify color matches gravity state

### SLIDE 40: MINI PLENARY

**Quick Assessment Questions**

1. What are the two possible values for boolean variables? (True and false)
2. How does if/else logic make decisions? (Checks condition, does different actions)
3. What visual feedback shows the gravity state? (Color change - blue/red)
4. Why does the box float when gravity is off? (No downward force)

**Check Your Understanding:**

* Can you create toggle behavior with boolean logic?
* How do visual cues help users understand system states?

### SLIDE 41: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 42: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Demonstrate gravity toggle with visual feedback
* Show box floating and falling based on state
* Explain how boolean logic controls physics

**Reflection Questions:**

* How does boolean logic make objects "smart"?
* What makes toggle systems useful for control?
* How do visual cues improve user experience?

**Next Lesson Preview:** Building a Mars control room to manage multiple objects

## LESSON 7: MARS CONTROL ROOM WITH MULTIPLE OBJECTS (7 SLIDES)

### SLIDE 43: LESSON TITLE

**Mission to Mars - Lesson 7: Mars Control Room with Multiple Objects**

**Learning Objectives:**

* Organize multiple objects using lists
* Create functions for reusable code blocks
* Control gravity for multiple objects simultaneously
* Build efficient control systems for complex environments

**Key Computing Concepts:** Lists, Functions, Code Organization, Multiple Object Control

### SLIDE 44: YOUR MISSION

**Your Mission Today**

Build a Mars control room using **lists** and **functions** to control multiple objects simultaneously. You'll create a gravity control system where pressing one button turns gravity off for all objects (making them float), and another button turns gravity back on (making them fall).

**Success Looks Like:** A control room where multiple objects respond to gravity changes together, demonstrating how functions and lists organize complex behaviors.

### SLIDE 45: CONCEPT EXPLANATION

**Understanding Lists and Functions for Organization**

**Lists:**

* Collections of objects for group management
* Efficient way to control multiple items
* Add objects once, affect them all together

**Functions:**

* Reusable code blocks for organizing actions
* Reduce repetition and improve organization
* Can be called multiple times

**Real-World Connection:** Mars mission control centers use automated systems to manage multiple spacecraft systems simultaneously

### SLIDE 46: CORE ACTIVITY

**Step-by-Step Control Room Creation**

**Step 1: Creating Object Lists**

when play clicked

create empty list gravityItems

add Crate to gravityItems

add Tool to gravityItems

add Equipment to gravityItems

**Step 2: Gravity Control Functions**

define function turnGravityOff()

set gravity pull to 0

for each element in gravityItems

set color of element to blue

define function turnGravityOn()

set gravity pull to 10

for each element in gravityItems

set color of element to red

**Step 3: Button Control System**

when GravityOff is clicked

call function turnGravityOff()

when GravityOn is clicked

call function turnGravityOn()

### SLIDE 47: MINI PLENARY

**Quick Assessment Questions**

1. How do you add objects to a list? (Add [object] to [list name])
2. What makes functions reusable? (Can be called multiple times)
3. How do you affect all objects in a list? (For each element loop)
4. Why organize code with functions? (Reduces repetition, improves clarity)

**Check Your Understanding:**

* Can you control multiple objects efficiently?
* How do lists make managing groups easier?

### SLIDE 48: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 49: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Demonstrate gravity control affecting multiple objects
* Show organized code using functions and lists
* Explain efficiency benefits of your control system

**Reflection Questions:**

* How do functions make your code better organized?
* What advantages do lists provide for multiple objects?
* How does your control room simulate real Mars systems?

**Next Lesson Preview:** Integrating all skills to create a complete Mars base

## LESSON 8: MARS BASE INTEGRATION (7 SLIDES)

### SLIDE 50: LESSON TITLE

**Mission to Mars - Lesson 8: Mars Base Integration**

**Learning Objectives:**

* Integrate all learned programming concepts
* Create complex interactive environments
* Apply positioning, animation, controls, and logic together
* Design creative solutions for Mars base systems

**Key Computing Concepts:** Integration, Project Planning, Complex Systems, Creative Application

### SLIDE 51: YOUR MISSION

**Your Mission Today**

Combine all learned skills to create an interactive **Mars base** with **camera tours** and **simple interactions**. You'll integrate positioning, animation, controls, and functions into one complete project. By the end of this lesson, you should have a working Mars base that showcases all your coding skills.

**Success Looks Like:** A complete Mars base demonstrating integration of all previous concepts with guided tours and interactive elements.

### SLIDE 52: CONCEPT EXPLANATION

**Understanding System Integration**

**Integration Concepts:**

* Combining multiple programming skills
* Using positioning, animation, controls, and logic together
* Creating complex interactive environments

**Project Planning:**

* Design before coding
* Plan how different systems work together
* Test components before combining

**Real-World Connection:** Real Mars bases will require integrated systems for life support, research, communication, and exploration

### SLIDE 53: CORE ACTIVITY

**Step-by-Step Mars Base Creation**

**Step 1: Environment Setup**

* Use Week 1 skills: position and scale base structures
* Apply realistic Mars environment design
* Plan base layout and key areas

**Step 2: Animation Integration**

* Use Week 2-4 skills: create moving elements
* Add simple camera tour path
* Include rotating equipment or moving vehicles

**Step 3: Interactive Elements**

* Use Week 5-6 skills: add button controls and toggles
* Include gravity control systems
* Add responsive user interactions

**Step 4: Function Integration**

* Use Week 7 skills: organize complex behaviors
* Apply functions to manage multiple interactive elements
* Test all systems working together

### SLIDE 54: MINI PLENARY

**Quick Assessment Questions**

1. Which skills from previous weeks are you combining? (All: positioning, animation, controls, logic, functions)
2. How do you plan complex projects? (Design first, test components, integrate gradually)
3. What makes a Mars base realistic? (Multiple functional areas, interactive systems)
4. Why is integration more challenging than individual skills? (Multiple systems must work together)

**Check Your Understanding:**

* Can you combine multiple programming concepts effectively?
* How do you troubleshoot when complex systems don't work?

### SLIDE 55: CHALLENGE CARDS

**See separate challenge cards for independent work**

**MILD Challenge Available** **MEDIUM Challenge Available** **HOT Challenge Available**

*Pick up your challenge card from the teacher*

### SLIDE 56: FINAL PLENARY

**Prove It Tasks and Reflection**

**Prove It:**

* Present your complete Mars base with guided tour
* Demonstrate integration of all learned concepts
* Explain design decisions and problem-solving approaches

**Final Unit Reflection Questions:**

* Which programming concept was most transformative for your projects?
* How did your problem-solving skills develop throughout the unit?
* What would you create next using these coding skills?
* How do these skills connect to real Mars exploration technology?

**Computing Vocabulary Mastery:** 3D Positioning, Animation Paths, Forever Loops, Parallel Processing, Event-Driven Programming, Boolean Variables, If/Else Logic, Lists, Functions, System Integration