Mission to Mars - Coding Unit (Year 5)

Duration: 6 weeks | **Age:** 10-11 years | **Platform:** CoSpaces Edu

---

Lesson 1: Solar System Animation

Learning Objectives

- Create smooth orbital animations using paths and forever loops
- Use parallel processing to make multiple objects move simultaneously
- Apply realistic timing so closer planets move faster than distant ones

Lesson Input

Key Concepts: Paths, Forever loops, Parallel processing **Code Structure:**

```
when play clicked
run parallel
  forever
    move Earth on path earth_path forward in 10 sec
run parallel
  forever
    move Moon on path moon_path forward in 3 sec
```

Building Your Animation:

1. Create circular paths for each object
2. Set different durations (closer planets = shorter times)
3. Use "run parallel" so all objects move at once

Mini Plenary

✓ Do your planets move continuously without stopping?
✓ Are closer planets moving faster than distant ones?
✓ What happens if you change the duration numbers?

Challenge Levels

Foundation: Sun-Earth-Moon system (3 objects)

- Create simple circular paths around the sun
- Earth orbits sun, Moon orbits Earth
- Success: Smooth continuous movement with no stopping

Intermediate: Inner solar system (5-6 planets)

- Add Mercury, Venus, Mars to your system
- Create logical speed progression (Mercury fastest, Mars slowest)
- Success: All planets moving with realistic speed relationships

Advanced: Full solar system + rotation

- Include all 8 planets with researched orbital periods
- Add planet rotation using additional animation blocks
- Success: Complex animation system with creative enhancements

Wrap-up

Today you mastered **paths** and **forever loops** to create realistic solar systems. Next lesson: controlling objects with buttons and creating smart on/off systems!

Lesson 2: Astronaut Control

Learning Objectives

- Control object movement using velocity for realistic space physics
- Create boolean variables that store true/false information
- Build toggle systems using if/else conditional logic

Lesson Input

Key Concepts: Velocity, Boolean variables, Conditional logic **Code Structure:**

```
when Up is clicked
  push Astronaut up with velocity 1

// Boolean toggle example:
when box is clicked
  if gravity_on = false
    set gravity_on to true
    set gravity pull to 10
  else
    set gravity_on to false
    set gravity pull to 0
```

Understanding Velocity:

- Velocity = force + direction that moves objects

- Higher numbers = faster movement
- Always include a "stop" button (velocity 0)

Mini Plenary

✓ Can you move your astronaut in all 6 directions?
✓ Does your toggle system change color when clicked?
✓ What happens when you change velocity numbers?

Challenge Levels

Foundation: 6-direction movement + stop

- Up, down, left, right, forward, backward buttons
- Include essential stop button (velocity 0)
- Success: All directions work smoothly with stop control

Intermediate: Multiple objects with different speeds

- Control 2-3 different objects (astronaut, tools, equipment)
- Experiment with velocity values (1, 2, 3) to see differences
- Success: Clear understanding of how velocity affects speed

Advanced: Boolean toggle system

- Create variables that remember on/off states
- Use if/else logic to switch between true and false
- Add visual indicators (color changes)
- Success: Objects change behavior and appearance based on conditions

Wrap-up

You mastered **velocity control** and **boolean logic**. Next: organizing your code with functions and lists to control multiple objects efficiently!

Lesson 3: Gravity Control Room

Learning Objectives

- Organize multiple objects efficiently using lists and functions
- Create reusable code blocks to reduce repetition
- Implement collision physics so objects bounce realistically

Lesson Input

Key Concepts: Functions, Lists, Collision detection **Code Structure:**

```
create empty list gravity_items
add [Light Object] to gravity_items
add [Heavy Object] to gravity_items

define function toggle_gravity()
  for each element in gravity_items
    if gravity_on = true
      set gravity pull to 10
    else
      set gravity pull to 0
```

Building Your Control Room:

1. Create walls with collision enabled
2. Add objects with different weights
3. Test that objects bounce off boundaries

Mini Plenary

✓ Do your objects bounce off walls instead of passing through?
✓ Does your function affect multiple objects at once?
✓ Can you see differences between light and heavy objects?

Challenge Levels

Foundation: Simple functions with lists

- Control multiple objects together using lists
- Create basic function that affects all listed objects
- Success: Function affects all objects, demonstrates code reusability

Intermediate: Physics with different weights

- Set realistic masses: Light (0.1-0.5kg), Heavy (5-10kg)
- Configure appropriate bounciness for each weight
- Success: Clear differences in how objects behave based on weight

Advanced: Boolean functions with collision logic

- Combine Lesson 2 toggle systems with list management
- Create functions that use conditional logic for multiple objects

- Success: Sophisticated toggle systems affecting multiple objects

Physics Settings

- **Light objects:** Bounciness 0.6-0.8
- **Heavy objects:** Bounciness 0.1-0.2

Wrap-up

You organized code with **functions** and **lists**, plus added realistic **collision physics**. Next: combining everything to build incredible structures on Mars!

Lesson 4: Mars Base Design

Learning Objectives

- Integrate all previous coding skills into a complex interactive environment
- Create guided camera tours that showcase different areas effectively
- Design realistic 3D spaces that tell a story about Mars exploration

Lesson Input

Key Concepts: Integration, Camera paths, Interactive objects **Code Structure:**

```
// Camera tour system
when Tour_Button is clicked
  move Camera on path tour_path forward in 20 sec

// Interactive elements
when Airlock_Door is clicked
  call function open_door()
  wait 3 sec
  call function close_door()
```

Designing Your Mars Base:

1. Plan different areas (habitat, laboratory, garage)
2. Create realistic Martian environment (red landscape)
3. Apply previous lessons: camera paths, button controls, physics

Mini Plenary

✓ Does your camera tour show all base areas clearly?
✓ Do interactive elements respond correctly to clicks?

✓ Does your base look and feel like a real Mars mission?

Challenge Levels

Foundation: Basic base (2-3 areas)

- Simple structures: dome habitat, landing pad, rover
- Single camera tour path (10-15 seconds)
- Success: Working camera movement, basic interactions

Intermediate: Interactive base (4-5 areas)

- Multiple specialized areas: habitat, lab, greenhouse, communications
- Longer camera tour with multiple stopping points (20-25 seconds)
- Success: Complex camera paths, multiple interactive features

Advanced: Complex systems (6+ areas)

- Comprehensive base with specialized zones and realistic details
- Multiple camera tour options (science tour, living areas tour)
- Success: Sophisticated integration of all concepts, innovative features

Wrap-up

You built incredible **Mars bases** by combining animation, control systems, physics, and interactions. Next: making your base educational with smart quizzes!

Lesson 5: Educational Quizzes

Learning Objectives

- Create educational content that responds intelligently to user answers
- Implement conditional logic to provide appropriate feedback
- Design user input systems that are intuitive and engaging

Lesson Input

Key Concepts: Conditional statements, User input, String comparison **Code Structure:**

```
when Quiz_Button is clicked
  if text of Answer_Input = "carbon dioxide"
    say "Correct! Mars atmosphere is 95% CO2"
    show object Reward_Item
  else
    say "Not quite - think about greenhouse gases!"
```

Building Educational Interactions:

1. Choose interesting Mars facts for your questions
2. Plan helpful hints for incorrect answers
3. Create positive responses for correct answers

Mini Plenary

✓ Do correct answers give positive, informative feedback?
✓ Do incorrect answers provide helpful hints?
✓ Are your Mars facts accurate and interesting?

Challenge Levels

Foundation: Simple yes/no questions

- 2-3 basic Mars questions with correct/incorrect responses
- Clear feedback that teaches something new
- Success: Basic conditional logic works, educational content is accurate

Intermediate: Multiple choice questions

- 3-4 questions with specific feedback for different answers
- Include scoring system to track progress
- Success: Sophisticated feedback system, multiple question types

Advanced: Complex educational system

- Multiple question formats (text input, multiple choice, true/false)
- Adaptive hints that get more specific if wrong
- Success: Comprehensive learning experience with innovative interactions

Sample Mars Quiz Topics

- "What gas makes up 95% of Mars atmosphere?"
- "How much would you weigh on Mars compared to Earth?"
- "What's the average temperature on Mars?"

Wrap-up

You created **interactive educational systems** using **conditional logic**. Final lesson: showcasing your incredible Mars missions!

Lesson 6: Showcase & Assessment

Learning Objectives

- Document and communicate your coding achievements using technical vocabulary
- Provide constructive feedback to peers using specific success criteria
- Reflect on problem-solving strategies and learning growth

Activities & Assessment Focus

1. Project Presentation (5 minutes each)

- Record 2-3 minute Mars base tour highlighting key features
- Explain coding concepts used
- Show most creative or challenging element

2. Peer Assessment

- Use structured feedback forms
- Identify successful coding concepts in others' projects
- Suggest one improvement

3. Reflection Questions

- Which coding concept was most challenging? Why?
- How did you debug problems when code didn't work?
- What would you add with unlimited time?

Mini Plenary

✓ Can you explain your code using proper technical vocabulary?
✓ Can you identify specific coding concepts in others' projects?
✓ What evidence shows your problem-solving skills?

Assessment Levels

Foundation: Core functionality demonstrated

- Basic animation, movement, and interaction systems work correctly

- Can explain simple coding concepts
- Success: Fundamental skills mastered, project functions as intended

Intermediate: Multiple features integrated creatively

- Advanced interactions, thoughtful design choices
- Uses technical vocabulary to explain complex systems
- Success: Strong integration of multiple concepts with creativity

Advanced: Sophisticated systems with innovation

- Complex code organization, innovative features beyond requirements
- Masterful use of technical vocabulary
- Success: Exceptional creativity combined with technical mastery

Final Wrap-up

Incredible Achievement! You've mastered **8 major coding concepts** and applied them creatively to build educational Mars exploration experiences.

Your coding superpowers: Creating moving systems, controlling objects, organizing complex code, simulating physics, building educational interactions, and solving problems independently.

Next adventures: Use these skills in science projects, design challenges, storytelling, and any subject where interactive systems can enhance learning!

Technical Vocabulary Checklist

By the end of this unit, students should understand:

- **Paths** - Routes objects follow
- **Velocity** - Force that moves objects
- **Boolean** - True/false variables
- **Functions** - Reusable code blocks
- **Lists** - Collections of objects
- **Collision** - Objects bouncing off boundaries
- **Conditional** - If/else decision making
- **Loops** - Repeating code blocks