Don't Spill the Liquid! - The Steady Hand Game

Put your steady hand to the test and learn about compound logic. Use the 'AND' operator to program a balancing game that needs two conditions to be true at the same time.

Courses

- Grades 6-12

Materials

- Cell phone and computer
- Internet connection
- Cardboard, scissors and a couple of rubber bands to hold the phone against the cardboard

Description

This activity introduces students to compound conditionals through the "AND" logical operator. They will build a balancing game that requires two sensor conditions (tilt on the X axis and tilt on the Y axis transformed into LED positions) to be true at the same time in order to succeed. This teaches a more advanced and more realistic way of making decisions in programming.

Educational Objectives

- Understand and apply the "AND" logical operator.
- Build compound conditional statements that evaluate several conditions.
- Solve a problem that requires checking several inputs at the same time.
- Apply boolean logic to an interactive game scenario.

Start (10 minutes) - The Power of "AND"

1. Welcome the class: **"Today, our programs will learn to multitask when they make decisions. We are going to build a game that will test your steady hand and your concentration."**
2. Introduce the concept with an analogy: **"To get into a high-security area, you might need a key card AND a retina scan. One of them alone is not enough, you need both. In programming, we use the 'AND' operator for these strict situations."**

3. Connect it to the game: **"To avoid 'spilling the liquid' in our game, you'll have to keep your balance from left to right AND from front to back at the same time. Let's see how to program this double condition!"**

Decisions with Multiple Requirements

In real life, a single question often isn't enough. For a self-driving car to move through an intersection, the light has to be green **AND** there must be no pedestrians crossing. To buy an item in a game, you need to have enough gold **AND** have room in your inventory. Life is full of decisions that depend on several factors that all have to be true at the same time.

The 'AND' Logical Operator

The **"AND"** operator is the tool we use to combine several questions. It's like a very strict security guard who asks for two forms of ID. He will only let you through if the first one is valid **AND** the second one is valid too. If either of the two fails, the whole condition counts as false. In boolean logic: - `TRUE` AND `TRUE` = **TRUE** - `TRUE` AND `FALSE` = **FALSE** - `FALSE` AND `TRUE` = **FALSE** - `FALSE` AND `FALSE` = **FALSE**

The Logic of Balance

For our game, the logic for staying in the "safe" zone is a compound condition: **IF** (`the X tilt transformed into LED positions is between LEDs 3 and 5`) **AND** (`the Y tilt transformed into LED positions is between LEDs 3 and 5`) **THEN**... you're safe. **ELSE**... alarm! If you lean too far forward (the Y condition fails) or too far to the left (the X condition fails), the overall result of the "AND" will be FALSE and you'll lose your balance.

Development (20-30 minutes) - Programming the Balance

1. Now that students understand why we need to check two things at once, it's time to build the game.
2. Guide them through **the instructions for creating the balancing game, paying special attention to how the 'AND' block is built by**

nesting the two conditions inside it, as detailed below. Encourage them to see just how hard it is to keep both conditions true at the same time.

Closing (7-10 minutes) - The Strict Logic of "AND"

1. Once everyone is busy trying to beat the game, take a moment to reflect on the logic that drives it.
2. Start the discussion: **"Let's take our condition apart. If you keep a perfect balance from left to right, but you lean far forward, why does the alarm go off? Because the 'AND' operator is very strict. In what other real-life situations do two or more conditions have to be met at the same time?"**

Reflect

What does the "AND" operator do? Why did we need it for this game instead of a simple "if"?

Describe a situation in which the alarm would go off. What values would `ledX` and `ledY` have to have for that to happen — and which way would you have to tilt the phone to make it happen?

If you wanted to make the game easier, would you widen or narrow the range of safe numbers (which is currently between 3 and 5)?