Build a Sensor(y) Smart Garden

Year level band: Years 5 - 6 and Years 7 & 8

Description: How often during the school holidays over summer do the plants in the school or home garden die from lack of attention and water because there is no one available to water them?

This is a particular problem during the vigorous growing season for garden vegetables in a school kitchen garden. Just when the plants need the most attention those who water and care for them are absent!

Imagine if you could build a soil moisture sensor that will automatically turn on a tap when the soil becomes dry and then turn off when the correct moisture level is achieved?

In this lesson, students explore this solution using a BBC micro:bit and a moisture sensor to solve this problem.

Resources:

- BBC Micro:bit with battery. (Available in CSER Lending Library Kit)
- 2 x Crocodile clips with cables
- Two long nails or screws
- Access to garden or garden resources (pots, soil, dirt).

Prior Student Learning: Some understanding of how a computer gives and receives instructions would be helpful but not necessary in first instance. An introduction to visual language programming such as scratch or blockly would be helpful too.

Summary

This project involves students learning about user-interface design and the importance of considering the user when designing digital solutions, and that user-interface design is about considering the user's interaction with a solution and how to meet those needs. In doing so, students develop skills in designing a solution for a user with specific needs, and being able to communicate their design intentions - with sketch designs, as well as verbally by sharing their designs with peers.

By reflecting on their own designs, as well as other designs, students develop skills in being able to evaluate designs and provide constructive feedback.

Year	Content Descriptors
5 - 6	Implement digital solutions as simple visual programs involving branching, iteration (repetition), and user input (ACTDIP020 - Scootle)
	Explain how student solutions and existing information systems are sustainable and meet current and future local community needs (ACTDIP021 - Scootle)
7-8 (introduction)	Define and decompose real-world problems taking into account functional requirements and economic, environmental, social, technical and usability constraints (ACTDIP027 - Scootle) Can also be done using general purpose programming language.

Element	**Summary of tasks**
Learning hook	If the school has a kitchen garden or a classroom plant that needs attention, have the children come up with a design solution that might provide the plant or garden with the care it needs in the summer holidays. Look at real-world problems where digital solutions have been developed through the use of digital technologies. Collect display, analyse and interpret data for the purpose of solving real-world problems. What is our problem? It's not necessarily *how* to water the garden but *when* is the most appropriate time to water the garden. Can we automate the process so that the garden is only watered when it is needed?
Achievement Standards Learning Map (Sequence)	Students incorporate user interface design into their designs and implement their digital solutions, including a visual program. They explain how information systems and their solutions meet needs and consider sustainability. Students manage the creation and communication of ideas and information in collaborative digital projects using validated data and agreed protocols. ● students consider their garden designs and review what is it that makes up a system. eg soil, moisture, sunlight, people... ● Use a micro:bit and sensors to monitor the moisture content of a garden. ● Students follow the examples in the micro:bit tutorial (https://makecode.micro:bit.org/projects/plant-watering) then look to see how it might be modified for example to scare away unwanted visitors. eg maybe the sound of an alarm. ● Students test and evaluate their own and one another's finished projects to see if the plants are watered when moisture content in the soil is reduced.
Learning input	The teacher works with the students to talk about the problems associated with looking after a garden or farm animals like chickens when a period of absence is expected. For example the Summer holidays. Identify the Problem: How do I keep the garden/plant watered? Introduce the design cycle: Empathise, Define, Ideate, Prototype and Test. (Repeat until solution found) *Empathy phase* How do I make sure plants don't die over Christmas?

	Systems Thinking ~What are all the things that affect my garden and water use? *Refine the problem*: How do I manage water use to assist my plants to grow? *Design Thinking;* Come up with ideas (Ideation) to solve the problem. *Computational Thinking*. ● Decomposition - Break down the problem and look first at how to measure moisture. ● Abstraction - Ignore other factors for now. ● Algorithms - Design sequences of instructions for the solution. ● Data Representation - What data is needed and how do we represent the data? Introduce the micro:bit and its capabilities. Have students work through tutorial https://makecode.micro:bit.org/, if new to this device.
Learning construction	The class are introduced to the micro:bit and the two ways they might be programmed either in a visual (5 -6) or text-based language (years 7-10) Students research the problem as commenced in the learning input discussions. Look at the different ways to measure moisture content in soil. By touch, by using conductive items eg nails or an especially designed moisture probe. The introductory sessions to this problem could take two to three lessons until students feel confident with the visual programming interface. Follow instructions in micro:bit tutorial. https://makecode.micro:bit.org/projects/plant-watering which contains the code for years 5-6. For years 7-10, discuss the sequence of instructions using the visual code as template and then implement in text-based. Provide pots of soil and moisture spray to adjust soil wetness for testing.
Learning demo	Students set up plants and soil with probes to test inside before taking probes outside to test in their garden if one is available. A battery pack is required here. Students share their code and design for their micro:bit moisture tester with their peers. Those who have mastered the capabilities of the micro:bit will be able to then plan how to add a pump that is activated by the soil drying in the pot or garden. A follow-up challenge for the students would be to show them some of the ways a servo or pump might be added to the micro:bit to allow water to be added to the soil until the analog reading shows a predetermined number turns off the pump.
Learning reflection	Students reflect on their initial designs and the feedback they received, and reflect on how they used this feedback to inform their final design. They reflect on the progress that they have made in comparison to their

	first design, and how their learning experience has informed their understanding of designing for diverse users.

Assessment:

In this lesson, teachers could **collect** evidence of learning and progression by the form of their artefacts, including **design** documents, **presentation** feedback, presentation recordings, photos of the final product and development stages.

	Quantity of knowledge			**Quality of understanding**	
Criteria	**Pre-structural**	**Uni-structural**	**Multi-structural**	**Relational**	**Extended abstract**
User-interface design	Standard controller - no consideration of the user.	A simple controller that explains general design considerations, but not necessarily unique to their user.	A controller with consideration made toward the user, as explained through a feature.	A controller that has considered the user through two or more design features supported by justification.	A controller that has addressed multiple user needs, with multiple features, and has a high level of complexity and justification for design features.
Design	No design used.	Basic design with no features identified.	Basic design with some features identified, but not linked to design justification.	Detailed design with numerous features identified and linked to design justifications.	Detailed design that brings in prior learning and/or independent learning beyond the task and possibly includes requirements, specifications, constraint factors.
Language	When describing their interface, no specific vocabulary is used.	The terms ‘controller’’ may be used as a general description.	The terms user-interface is used as a general description.	The terms user-interface is used confidently with specific reference to learner’s work.	Specific vocabulary like ‘requirements’, ‘specifications’ and ‘constraints’ is used, going beyond the set language.

Teacher/Student Instructions:

Introduction to the micro:bit https://www.youtube.com/watch?v=Wuza5WXiMkc

https://makecode.micro:bit.org/

https://makecode.micro:bit.org/projects/plant-watering

CSER Professional Learning:

This lesson plan corresponds to professional learning in the following CSER Digital Technologies MOOCs:

F-6 Digital Technologies: Foundations

- Unit 4: Digital Systems

See: http://csermoocs.adelaide.edu.au/moocs

Further Resources:

- Bob Elliott: tutorials on building smart gardens with the BBC micro:bit
  http://smartgarden.strikingly.com/

- Martin Levins smart garden solutions - An explanation on problem-solving.
  - https://youtu.be/TrjCLS-W7aM
  - https://youtu.be/3f3AZb9RZlU
  - https://youtu.be/WTOB7Bpz6Pk
  - https://youtu.be/gQrZAq-o0j4

- micro:bit tutorials and projects
  https://makecode.micro:bit.org/

- Building a water sensing probe with the BBC micro:bit - Tutorial
  https://makecode.micro:bit.org/projects/soil-moisture

Author: Peter Lelong