

Healthy oceans

**Activities:** 2

**Programming languages:** MakeCode

**Target age:** 11-14 yrs, 14-16 yrs

**Subjects & topics:**

* Computer systems: Sensors, Input/output
* Computational thinking: Algorithms, Flowcharts
* Networks: IoT
* Design & technology: Electronics, Product design
* Global Goals: 14 Life below water

# Design challenge summary

Two longer activities to build prototypes to learn about how wireless networks, smart materials and algorithms can help keep our oceans clean.

Each activity could last for one or more extended sessions in a formal lesson, code club or maker space context.

In the **Ocean health monitor** project, students make prototypes of wireless sensors to monitor the environment at sea.

In the **Oil spill cleaner-upper** project, students design an efficient algorithm to clean an area of sea in the first part, then create an autonomous vehicle that can mop up oil spills.

## Overall key learning

* Understand what the Global Goals are
* Understand what [Global Goal 14](https://www.globalgoals.org/14-life-below-water) is and its significance
* Understand the basics of transmitting data
* Produce a data node product to meet the success criteria
* Produce an Oil Spill Cleaner-Upper product to meet the success criteria
* Develop the products further with additional features

## Additional skills

Design thinking, prototyping, modifying, testing, problem-solving, creative thinking

## Activity 1: Ocean health monitor

Make working prototypes of wireless sensors which could be used to monitor the environment at sea, including waves and weather and learn about how real-world sensors are connected through gateways to the internet.

**Key learning:**

* Understand what the Global Goals are
* Understand what [Global Goal 14](https://www.globalgoals.org/14-life-below-water) is and its significance
* Understand the basics of transmitting data
* Produce a data node product to meet the success criteria
* Develop the product further with additional features

## Activity 2: Oil spill cleaner-upper

This activity is in two parts, you can do either part – or both. The first part can be completed with no additional hardware or even micro:bits, comprising a challenge to design an efficient algorithm to clean an area of sea. The second part builds on this to create an autonomous vehicle that can mop up oil spills.

**Key learning:**

* Understand what the Global Goals are
* Understand what [goal 14](https://www.globalgoals.org/14-life-below-water) is and its significance
* Produce an Oil Spill Cleaner-Upper product to meet the success criteria
* Develop the product further with additional features

# Curriculum links

## England National Curriculum

#### KS3 computing curriculum

Curriculum aims:

* can understand and apply the fundamental principles and concepts of computer science, including abstraction, logic, algorithms and data representation
* can analyse problems in computational terms, and have repeated practical experience of writing computer programs in order to solve such problems

Students should be taught to:

* design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems
* use logical reasoning to compare the utility of alternative algorithms for the same problem
* make appropriate use of data structures [for example, lists, tables or arrays];
* understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems
* understand how data of various types (including text, sounds and pictures) can be represented and manipulated digitally, in the form of binary digits
* undertake creative projects that involve selecting, using, and combining multiple applications, preferably across a range of devices, to achieve challenging goals, including collecting and analysing data and meeting the needs of known users
* create, reuse, revise and repurpose digital artefacts for a given audience, with attention to trustworthiness, design and usability
* develop their capability, creativity and knowledge in computer science, digital media and information technology
* develop and apply their analytic, problem-solving, design, and computational thinking skills

[Read the full KS3 computing curriculum](https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/239067/SECONDARY_national_curriculum_-_Computing.pdf)

#### KS3 DT curriculum

Students should be taught to:

* design - develop specifications to inform the design of innovative, functional, appealing products that respond to needs in a variety of situations
* design - develop and communicate design ideas using annotated sketches, detailed plans, 3-D and mathematical modelling, oral and digital presentations and computer-based tools
* make - select from and use a wider, more complex range of materials, components and ingredients, taking into account their properties
* evaluate - investigate new and emerging technologies
* evaluate - test, evaluate and refine their ideas and products against a specification, taking into account the views of intended users and other interested groups
* evaluate - understand developments in design and technology, its impact on individuals, society and the environment, and the responsibilities of designers, engineers and technologists
* technical knowledge - understand how more advanced electrical and electronic systems can be powered and used in their products [for example, circuits with heat, light, sound and movement as inputs and outputs]
* apply computing and use electronics to embed intelligence in products that respond to inputs [for example, sensors], and control outputs [for example, actuators], using programmable components [for example, microcontrollers].

[Read the full KS3 DT curriculum](https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/239089/SECONDARY_national_curriculum_-_Design_and_technology.pdf)

## Scotland Curriculum for Excellence

#### Technologies

* I can informally compare algorithms for correctness and efficiency. (TCH 4-13b)
* I can explain the overall operation and architecture of a digitally created solution (TCH 4-14b)

[Read the full Curriculum for Excellence: technologies](https://education.gov.scot/Documents/Technologies-es-os.pdf).

## Northern Ireland Curriculum

#### Science and technology - technology and design - statutory requirements, KS3

* Design – identifying problems; investigating, generating, developing, modelling and evaluating design proposals; giving consideration to form, function and safety;
* Communication – use of free-hand sketching and formal drawing techniques and ICT tools (including 3D modelling);
* Manufacturing – selecting and using materials fit for purpose; safe use of a range of tools and processes appropriate to materials, demonstrating accuracy and quality of outcome;
* Explore technical inventions and designs that have met a social need cost-effectively.
* Identify product needs and pursue sustainable harmonious design solutions in a local outdoor/indoor context. Education for sustainable development
* demonstrate practical skills in the safe use of a range of tools, machines and equipment;
* research and manage information effectively to investigate design issues, using mathematics and ICT where appropriate;
* show deeper understanding by thinking critically and flexibly, solving problems and making informed decisions, using Mathematics and ICT where appropriate;
* demonstrate creativity and initiative when developing ideas and following them through;
* work effectively with others;

[Read the full technology and design statutory requirements](https://ccea.org.uk/downloads/docs/ccea-asset/General/Statutory%20Requirements%20for%20Technology%20and%20Design%20at%20Key%20Stage%203.pdf)

#### Primary using ICT - desirable features - computational thinking and coding

**Level 5**

Pupils should:

* create more sophisticated coding projects using a broad range of commands and more than one platform; and
* solve a more complex problem using commands in a programming environment.

**Programmable devices (such as Parrot Drone, MicroBit or Sphero)**

* as a class look at and talk about examples of coding projects, including using multiple ‘if...then’ and ‘if...then...else’ commands, variables, sensors, events, operators and comparators;
* recognise how they can decompose these projects;
* in small groups, plan their own coding project, demonstrating a clear sense of purpose and audience, showing understanding of abstraction by deciding what details they need to include and what they can leave out, working out what different parts of the program must do and using logical reasoning to discuss and compare the commands that are required for their algorithm and predicting the outcome;
* use a range of commands to create a project, including variables, operators and control statements such as ‘if... then...’ alongside the use of ’if...then...else’ and comparators;
* test and debug at regular intervals and collaborate with others to solve problems as they arise;

**Finally**

* share their work (possibly using digital tools), respond to feedback, and comment on the work of others evaluating process and outcome; and
* organise files and publish work online (if available) so that others can view it.

[Read all Primary using ICT desirable features](https://ccea.org.uk/downloads/docs/ccea-asset/Curriculum/Primary%20Using%20ICT%20Desirable%20Features%20Update%202019.pdf)

## Curriculum for Wales

#### Science and technology

Progression step 4 - computation is the foundation for our digital world:

* I can select and use data structures that efficiently manage data in algorithms.
* I can select and use multiple sensors and actuators that allow computer systems to interact with the world around them.
* I can explain the techniques used to store and transfer data and understand their vulnerabilities.

Progression step 5 - computation is the foundation for our digital world:

* I can identify, define and decompose problems, choose appropriate constructs and express solutions in a variety of environments.
* I can test, evaluate and improve a solution in software.
* I can design and create physical systems that use appropriate components and logic to complete tasks and achieve goals.

[Read the full science and technology curriculum](https://hwb.gov.wales/curriculum-for-wales/science-and-technology/descriptions-of-learning/)

#### Digital competence framework

Progression step 4 - data and computational thinking - problem-solving and modelling:

* I can create a simple model or self-contained algorithm.
* I can identify the different parts of an algorithm to determine their purpose.
* I can detect and correct errors in algorithms.

Progression step 5 - data and computational thinking - problem-solving and modelling:

* I can independently create and design models, and explain how they represent real-world problems, e.g. selecting and correctly using an appropriate method for illustrating a problem, such as a flowchart or spreadsheet.
* I can develop logical solutions to determine the input, outputs and processes of a program, e.g. following pseudocode or a flowchart to come to an outcome, developing a written sequence of steps that could be followed.

[Read the digital competence framework](https://hwb.gov.wales/curriculum-for-wales/cross-curricular-skills-frameworks/digital-competence-framework)

## Code.org

#### CS Discoveries

Unit 1

Concepts included:

* problem solving
* inputs and outputs
* storing and processing information

Unit 4

Concepts included:

* social impact of computing
* understanding the needs of others when designing a solution
* team project
* testing and acting on feedback
* iteration

[Read the full Code.org CS Discoveries curriculum](https://studio.code.org/courses/csd-2021)

## USA CSTA Standards

#### Grades 6-8

* 2-CS-02 - Design projects that combine hardware and software components to collect and exchange data.
* 2-DA-08 - Collect data using computational tools and transform the data to make it more useful and reliable.
* 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms.
* 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
* 2-AP-17 - Systematically test and refine programs using a range of test cases.
* 2-AP-18 - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.

[Read the CSTA Standards in full.](https://csteachers.org/k12standards/ )

This content is published under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0)](https://creativecommons.org/licenses/by-nc-sa/4.0/) licence.