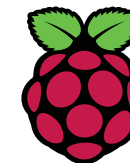






# Digital making curriculum



raspberrypi.org

	Design	Programming	Physical computing	Manufacture	Community and sharing
<b>Creator</b> 	Design basic 2D and 3D assets	Use basic programming constructs to create simple programs	Use basic digital, analogue, and electromechanical components	Use basic materials and tools to create project prototypes	Engage and share with the digital making community
<b>Builder</b> 	Combine 2D and 3D assets in the assembly of a project	Apply basic programming constructs to solve a problem	Combine inputs and/or outputs to create projects or solve a problem	Use manufacturing techniques and tools to create prototypes	Collaborate on digital making projects with other community members
<b>Developer</b> 	Use multiple designed assets in completed products and models	Apply abstraction and decomposition to solve more complex problems	Process input data to monitor or react to the environment	Use manufacturing techniques and tools to create a completed product	Support others in the design and build of their digital making projects
<b>Maker</b> 	Design multiple and integrating assets for use in complex finished projects and models	Apply higher-order programming techniques to solve real-world problems	Create automated systems to solve complex real-world problems	Independently use fabrication systems to produce complex finished projects	Educate others in the skills and ethos of digital making

## Design basic 2D and 3D assets.

Learners can use basic design principles to produce simple graphics, videos, circuit schematics, formatted web pages, cutting sheets, and 3D model components, utilising CAD and graphics software, or markup languages.

### Example outcome

- Can use CAD tools to create simple 2D designs for projects
- Can use a markup language to create and style text and images for a webpage
- Can use graphical applications to manipulate bitmap and vector images

### Example projects

- [Robot antenna \(Scratch, Raspberry Pi\)](#)
- [Pixel pet \(Python, Sense HAT\)](#)
- [Build a robot \(HTML/CSS\)](#)
- [Stickers \(HTML/CSS\)](#)
- [Cat meme generator \(HTML/CSS/JavaScript\)](#)

## Use basic programming constructs to create simple programs

Learners can write short programs demonstrating awareness of simple programming concepts, such as sequencing, repetition, variables, and selection.

### Example outcomes

- Can use simple control flow statements
- Can use variables and simple data structures
- Can use a variety of logical, arithmetic, and comparison operators

### Example projects

- [Lost in space \(Scratch\)](#)
- [Ghostbusters \(Scratch\)](#)
- [Chatbot \(Scratch\)](#)
- [Story time \(Python\)](#)
- [Turtle race \(Python\)](#)

## Use basic digital, analogue, and electromechanical components

Learners can write programs which use active and passive electronic components. They can receive data from input components and control output components using a computer or microcontroller.

### Example outcomes

- Can design and construct simple prototype circuits using components connected directly to GPIO pins
- Can recognise and use polar components, such as LEDs and capacitors
- Can write programs to read data from digital and analogue inputs and to control digital and analogue outputs

### Example projects

- [Whoopi cushion \(Python, Raspberry Pi\)](#)
- [Spinning flower wheel \(Python, Raspberry Pi, Explorer HAT\)](#)
- [Burping jelly baby \(Python, Raspberry Pi\)](#)
- [Frustration \(BBC micro:bit\)](#)

# Creator

## Use basic materials and tools to create project prototypes

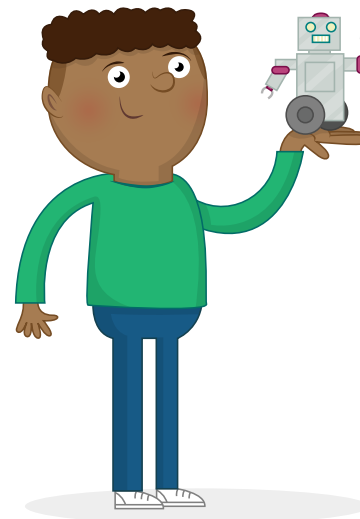
Learners can use tools to cut, measure, and join materials, and use bonding agents where appropriate, to produce simple parts for prototype projects. They can produce prototype circuits, following simple circuit schematics.

### Example outcomes

- Can create simple circuits, using a breadboard and jumper leads or crocodile clips
- Can use and manipulate upcycled materials for use in prototypes
- Can repurpose commercial products for use in project prototypes

### Example projects

- [Cress egg heads \(Python, Raspberry Pi, Pi Camera\)](#)
- [Tweeting Babbage \(Python, Raspberry Pi\)](#)
- [Whoopi cushion \(Python, Raspberry Pi\)](#)



## Engage and share with the digital making community

Learners interact with other members of the community, and share their creations. They may attend events and meetups, or interact with other community members online.

### Example outcomes

- Share a digital creation with the wider community
- Participate in an event within the wider community
- Engage in an online discussion about a digital making project
- Understand the importance of not sharing personally identifiable information online

### Example projects

- [Username generator \(Scratch\)](#)
- Use safe online sharing platforms to showcase their digital creations
- Attend a Raspberry Jam and listen to some talks or attend a workshop
- Visit their local makerspace and learn from others in the community

## Combine 2D and 3D assets in the assembly of a project

Learners can design 2D and 3D assets for use in projects, using CAD and graphics software or markup languages. They can independently produce simple graphics, video, circuit schematics, formatted web pages, cutting sheets, and 3D model components for a project.

### Example outcomes

- Can use simple video editing tools to manipulate footage
- Can use HTML, CSS, and JavaScript to produce browser-based content
- Can use CAD tools to produce component parts for 3D structures

### Example projects

- Create a sprite with an animated walk cycle for use in a computer game
- Create a website to showcase a digital making project
- Create a 3D model of a physical make to demonstrate its planned appearance and dimensions

## Manufacture

## Use manufacturing techniques and tools to create prototypes

Learners can choose and manipulate appropriate materials for use in their finished prototypes. They can use a variety of techniques and tools to accurately cut, measure, and manipulate materials. They can construct circuits by soldering to prototyping boards.

### Example Outcomes

- Can create soldered circuits using stripboards or protoboards
- Can use tools to manipulate materials and create component parts for a prototype
- Can use a multimeter to debug circuits

### Example Projects

- [Getting started with wearables \(C++, Adafruit FLORA\)](#)
- [See like a bat \(Python, Raspberry Pi\)](#)

## Apply basic programming constructs to solve a problem

Learners are able to develop algorithms that make use of basic programming constructs to satisfy predefined outcomes or project briefs.

### Example outcomes

- Can create subroutines/procedures/functions in their programs
- Can programmatically read and manipulate a data structure
- Can use standard communication protocols and APIs to transfer data between computers and applications

### Example projects

- [Dodgeball \(Scratch\)](#)
- [Create your own world \(Scratch\)](#)
- [Secret messages \(Python\)](#)
- [Turtley amazing \(Python\)](#)
- [Hamster party cam \(Python, Raspberry Pi, Camera Module, Explorer HAT\)](#)

## Combine inputs and/or outputs to create projects or solve a problem

Learners can write programs that use active and passive electronic components in combinations with one another. They can receive data from input components, perform limited data processing, and control multiple output components using a computer or microcontroller.

### Example outcomes

- Can use sensors to trigger real-world outputs, such as cameras and electromechanical devices
- Can use combinations of input and output components to transfer data between systems
- Can use transistors and relays to power components requiring higher voltages

### Example projects

- [Raspberry Pi Zero time-lapse camera \(Python, Raspberry Pi, Camera Module\)](#)
- [Making a laser tripwire with a Raspberry Pi \(Raspberry Pi\)](#)
- [GPIO Music box \(Python, Raspberry Pi\)](#)

## Community and sharing

## Collaborate on digital making projects with other community members

Learners collaborate on projects with other members of the community. This could be face-to-face or online, in either formal or informal settings.

### Example Outcomes

- Collaborate in a digital make with other people
- Participate in a digital making event with others in the community
- Be aware of data privacy rules and guidelines for the safe sharing of information online

### Example Projects

- [Password generator \(Python\)](#)
- Create a project with another member of their Code Club/CoderDojo/etc.
- Team up with some friends and go to a hackathon
- Find someone from the community who has made a project similar to theirs, and ask them for help in their make

# Builder



## Design

### Use multiple designed assets in completed products and models

Learners can design assets in a variety of media, and combine those assets into finished products. They can independently produce graphics, video, circuit schematics, interactive webpages, cutting sheets, and 3D models which meet a design brief.

#### Example outcomes

- Can use CAD software to design circuits and PCBs
- Can use CAD software to design 2D components to be laser cut and perfectly assembled
- Can use CAD software to design 3D models and components to be 3D printed and assembled

#### Example projects

- Create a 2D design for the components of a simple box to be used in a laser cutter
- Create files for a 3D body of a robot which can be used in a 3D printer
- Produce a design for a PCB which can be etched or sent for fabrication

## Manufacture

### Use manufacturing techniques and tools to create a completed product

Learners can use industry-standard prototyping and manufacturing techniques, such as 3D printing, laser cutting, and PCB etching, to produce and assemble components to be used in their finished projects.

#### Example outcomes

- Use cutting tools and joining/bonding techniques to create a finished product
- Solder components to PCBs and test circuits using a multimeter
- Can manufacture products which integrate with existing systems

#### Example projects

- [3D Printed Astro Pi flight case \(Raspberry Pi, Camera Module, Sense HAT, 3D printer\)](#)
- Use MDF and wood glue to produce a concealed housing for a burglar alarm
- 3D print a decorative ornament that contains flashing LEDs

## Programming

### Apply abstraction and decomposition to solve more complex problems

Learners are able to work on larger problems. They are able to simplify and break down larger problems, making use of abstraction and decomposition, as well as more complex reusable data structures.

#### Example outcomes

- Can decompose a large problem into parts and design algorithms to solve them
- Can recognise similar problems, and apply generic solutions and abstractions
- Can effectively combine functionality from multiple libraries or APIs and refer to documentation
- Can write code in a readable way and/or comments where necessary

#### Example projects

- [Dress for the weather \(Python\)](#)
- [Flappy astronaut \(Python, Raspberry Pi, Sense HAT\)](#)
- [Getting started with the Twitter API \(Python\)](#)

## Physical computing

### Process input data to monitor or react to the environment

Learners can use a computer or microcontroller to design and write programs to process data from external sources and to control output devices that react to, or alter, their environment. They can design and write programs to monitor various aspects of an environment.

#### Example outcomes

- Combine and process data from multiple sources to build a real-time system
- Use, and extrapolate values from, input data to make predictions about a system
- Use output devices to react to live data

#### Example projects

- [Parent detector \(Python, Raspberry Pi, Camera Module\)](#)
- [Sense HAT marble maze \(Python, Raspberry Pi, Sense HAT\)](#)
- [Sense HAT Minecraft map \(Python, Raspberry Pi, Sense HAT\)](#)

## Community and sharing

### Support others in the design and build of their digital making projects

Learners offer support and assistance to other members of the community. They create content which other community members can follow to create projects.

#### Example outcomes

- Assist others in a digital make, either online or in person
- Produce learning materials to guide others in a digital make
- Produce and maintain code that is shared through an online version control platform
- Contribute to a community project
- Give appropriate credit when making use of others' work

#### Example projects

- [Getting started with Git](#)
- Create a YouTube tutorial for their project
- Join their local makerspace and help others with their digital makes
- Give a talk at a developer conference, a Raspberry Jam, or a maker meetup

# Developer



## Design multiple and integrating assets for use in complex finished projects and models

Learners can design a range of assets in a variety of media, to be used in projects and models that solve real-world-problems. They can analyse a problem and then produce graphics, videos, circuit schematics, interactive web pages, cutting sheets, and 3D models that assist in solving that problem.

### Example outcomes

- Use PCB design tools to create PCBs that conform to appropriate dimensions and component positioning
- Use CAD tools to design housings and component parts which integrate into a finished product
- Use graphics and video editing tools to produce complete audiovisual products
- Give thought to user interaction and accessibility

### Example projects

- Produce a detailed character sheet for use in a graphical computer game
- Produce a 3D model of a character that can be imported into a game engine
- Produce a dynamic website that displays data harvested from an API

## Apply higher-order programming techniques to solve complex real-world problems

Learners are able to make use of a variety of programming paradigms, and combine data from various systems to solve complex, real-world problems.

### Example outcomes

- Can use a variety of programming paradigms in their programs, choosing techniques appropriate to the problem at hand
- Can implement and use a variety of more complex data structures in their programs, such as trees, graphs, and sets

### Example projects

- Create an importable module that interacts with the Minecraft API, allowing the easy construction and manipulation of large structures
- Create an automated maze generator for producing differently sized mazes that are suitable for displaying on an LED matrix
- Create a web-based front end for a HAB balloon, where sensor data received via radio is displayed in real time

## Create automated systems to solve complex real-world problems

Learners can analyse a real-world problem and design and create automated systems that monitor, react to, or influence an environment to solve the problem.

### Example outcomes

- Process multiple data and/or input sources, and use them to control multiple and interconnected output devices
- Use online data sources and inputs to control output devices
- Publish data from sensors on an online platform and display the data graphically

### Example projects

- Create a system which monitors and controls the environment of living things such as fish or plants
- Create a system which monitors local weather conditions and warns people of impending bad weather
- Produce a mechanical arm which can be monitored and controlled over the internet

## Community and sharing

## Manufacture

## Independently use fabrication systems to produce complex finished projects

Learners can appropriately combine a variety of industry-standard manufacturing techniques to independently produce and assemble components for a finished project.

### Example outcomes

- Use a 2D cutter to produce 2D components for assembly
- Use a 3D printer to produce a completed product or components for an assembly
- Use exposure and etching techniques to produce custom PCBs

### Example projects

- [Grandpa scarer \(Python, Raspberry Pi\)](#)
- Print a 3D housing for a Pi Zero Game Boy clone
- Produce a custom PCB for an automated pet feeder

# Maker



## Educate others in the skills and ethos of digital making

Learners take on an educator or mentor role to assist in the learning of other community members. They may take part in workshops and seminars, or inspire others through educator-focused volunteering.

### Example outcomes

- Regularly assist others in their digital making experiences
- Educate or assist large groups of individuals in their digital makes
- Regularly publish learning materials for use by the community
- Make code and data openly available (using an appropriate licence) for use by other community members
- Ensure all gathered data complies with local data protection laws

### Example projects

- Write tailored resources that meet the needs of a particular audience, e.g. children attending a Code Club
- Become a volunteer at Code Club, CoderDojo, or Pioneers
- Run a workshop at a Raspberry Jam or a conference

# Computational thinking

One of the aims of the Raspberry Pi Foundation is to help people to learn about computer science and how to make things with computers. We believe that learning how to create, control, and contrive technology will help people shape an increasingly digital world and prepare them for work of the future.

Computational thinking is at the heart of the learning that we advocate. It is the thought process that underpins computing and digital making: formulating a problem and expressing its solution in such a way that a computer can effectively carry it out. Computational thinking covers a broad range of knowledge and skills, including but not limited to:

- **Logical reasoning**
- **Algorithmic thinking**
- **Pattern recognition**
- **Abstraction**
- **Decomposition**
- **Debugging**
- **Problem solving**

By progressing through our curriculum, learners will develop computational thinking skills and put them into practice.

- A**
- abstraction** Representing a concept by omitting unnecessary details, focussing only on the aspects which are important
  - active sensor** Sends a signal and then detects changes in the environment based on the data returned, usually requires an external power source
  - algorithm** A series of precise instructions to solve a problem
  - analogue signal** A continuous variable signal
  - API** Application programming interface, a set of functions used when building software
  - assets** Items needed to create a project, such as images or 3D-printed parts
- B**
- bonding agent** Adhesive for sticking components together (e.g. glue)
  - breadboard** A reusable circuit board for prototyping electronic circuits
- C**
- CAD** Computer-aided design. Allows precise 2D and 3D drawings and models of real-world objects to be designed on a computer
  - circuit** A path along which current flows
  - computational thinking** A set of mental skills that help to define a problem in such a way that a computer can solve it
  - control flow** The order in which a computer executes instructions
- D**
- data structure** A format for storing and organising data (e.g. an array)
  - decomposition** Breaking a problem down into smaller, more manageable parts
  - digital signal** A signal that is either on or off
  - digital making** Creative projects requiring technical skill and understanding

# Glossary

- G** **GPIO** General-purpose input/output, pins connected to the Raspberry Pi which can be controlled with code
- I** **input** Data entered into a program (e.g. a mouse click)
- M** **markup language** A language specifying the presentation and style of text
- microcontroller** A small computer using a single chip
- O** **output** Data sent out by a program (e.g. sound from a speaker)
- P** **passive sensor** Detects changes in the environment without actively sending out a signal (e.g. a PIR sensor)
- PCB** Printed circuit board, a board connecting electronic components
- programming paradigm** A style or way of thinking about programming (e.g. object-oriented)
- protoboard** Allows you to add components to a prototype circuit by soldering or cutting strips
- R** **repetition** Executing the same code multiple times (while/for loops)
- S** **schematic** A diagram or plan representing a circuit
- selection** Choosing which code to execute based on a condition (if/else)
- sequencing** Executing lines of code one after another
- subroutine** A named set of instructions in a program (also referred to as a function/procedure)
- U** **upcycle** Using materials which otherwise would be thrown away in order to make something of value
- V** **variable** A named area in memory where data is stored