# Lesson 1 – What is micro:pyhon?

|  |  |
| --- | --- |
| The Big Picture – Why Is This Relevant? | Learning Objectives |
| This introductory lesson will take learners through what they will be learning for the next 40 weeks and will introduce some of the concepts around project based learning which ay be new to them. | * Learners will understand what the course involves, what they will be learning and why it is important * Learners will be familiar with the difference between the theory and project lessons * Learner will understand the context of programming physical devices and how it relates to careers in STEM |
| Engagement – How Can I Engage Learners? | Assessment for Learning |
| This is the first lesson of a large course and so learners should be interested in what is to come as well as what they will be doing. For some this may be their first experience of a textually derived language and this may motivate them. | **Expected Progress:**   * All learners will be familiar with the layout of the course and the difference between the theory and project lesson. Learners will understand what the ‘success criteria’ is for   **Good Progress:**   * Learners will understand the differences and similarities between the examples of block based and text based code   **Exceptional Progress:**   * Learners will confidently translate block based code examples into Python |
| Key concepts | Key words |
| * Code can be written in both text and blocks, they do the same thing but more can be done with actual code * Text based languages have a specific syntax that must be used, this is not as forgiving as blocks but the concepts are the same * Indentation really matters! * Case sensitivity | * Python * MicroPython * Text based language vs blocks * Indentation * Case sensitivity * Power cycle (turn it off and on again) |
| Differentiation | Resources |
| Some learners will be anxious about the transition to text based languages and so will require more support initially. It is typical for learners to struggle to translate the techniques that they may have been comfortable with in MakeCode into Python. | * Access to Mu or equivalent IDE |
| Lesson Flow | |
| * Introduce the course to the class and explain that it is a mix of theory, activities and project * Explain that the theory lessons will have tasks and activities and will always us the same format * Go through the activity sheet and explain what each section is for * Ensure learners understand what the success criteria are as this is vital throughout the course * Explain that the course has several projects that will be done in groups and that communication, collaboration and discipline will be required to complete the tasks successfully * Introduce the concept of a textually derived language and see if learners have heard of any others * Ensure that learners understand the difference between mark-up (HTML, CSS) and actual programming languages such as Python, C++, PHP etc. * Explain that the same things (and more) can be done in Python that can be done with blocks and look at the comparisons in the slides * Get learners to complete the activity sheet | |
| Making | |
| There is no making in this lesson | |