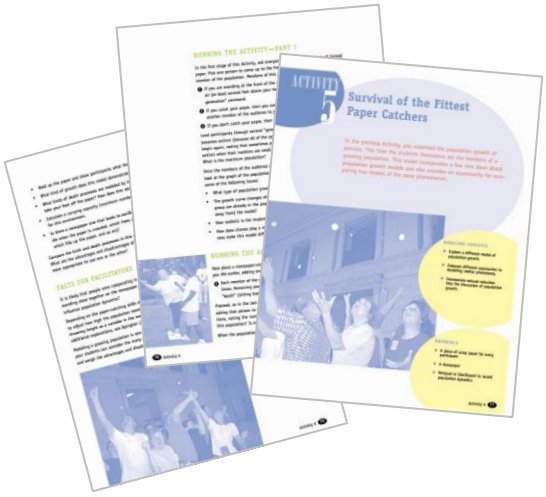
An outdoor activity to model population dynamics. A scaffolded project to develop a model of a simple eco-system, with extension challenges.

Preparation required:
Paper Catchers instructions, plentiful supply of old newspapers.
Sample Undersea World programs available to all.
Building A Marine Eco-system activity sheets.

Paper Catchers

Our final set of activities look at building a model eco-system. It starts with another kinaesthetic activity from 'Adventures in Modeling', designed to illustrate the cyclical nature of stable eco-systems. We'll then consider ways to set a final model building challenge.

The relevant chapter from the book is included in the resources. A 3 minute video from Project GUTS explains the idea well: youtu.be/Fywq_jKCDpY. You will require lots of sheets of paper to crumple, a large newspaper and some method for recording results. This is a fun activity, preferably done out of doors.

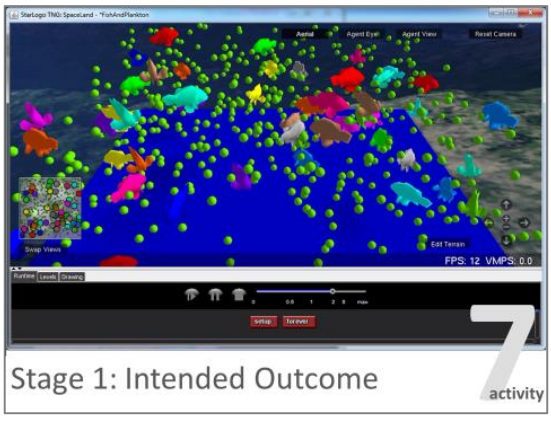


Over 3 rounds, students are introduced to the idea of negative feedback loops which impact on population dynamics. Apart from co-ordinating the activity, the key input from the teacher is to make explicit the role of the newspaper participants stand on. This is analogous to a finite food supply. Recording results, students will observe the cyclical fluctuations in the population that we observed in our predator – prey model in the last exercise.

The Undersea World

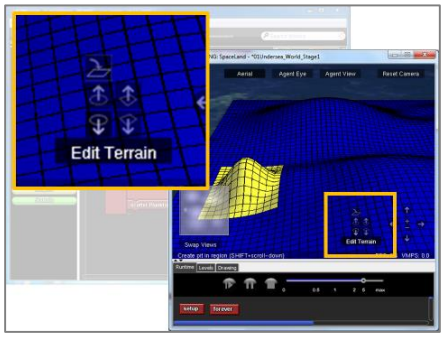
Our final activity is based on another model introduced by Project GUTS. Students will build a marine eco-system. It is tackled in two parts. The first stage is scaffolded, so the students don't start with a blank model. The second stage will allow them to develop the model in various ways. The video embedded in the presentation shows the intended outcome from the first stage of the project.

The model is similar to the examples shown at the start of the Unit. We're using StarLogoTNG. We have a model with our familiar two buttons. The undersea world has already been setup. There are a variety of species of fish, represented by the different colours. Their food source, plankton, are represented by the green balls. As the simulation runs we can see the familiar pattern of negative feedback in population dynamics. The food is eaten, it becomes scarce, and the population falls. The food regenerates. Once it is abundant the fish population similarly increases. Let's investigate what we give students as a starting point.

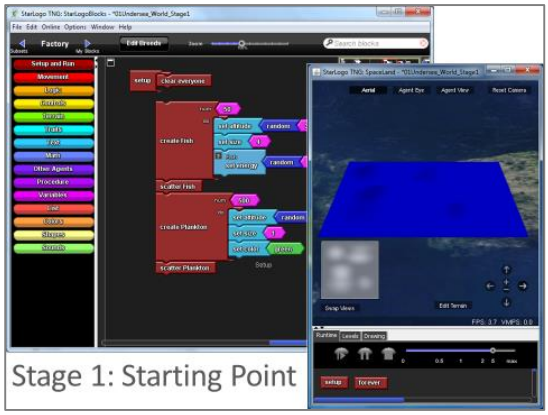


There are three sample programs provided. One is a complete version, two are incomplete. One of those has no comments to help students. We use one with comments; *01Undersea_World_Stage1.sltng*.

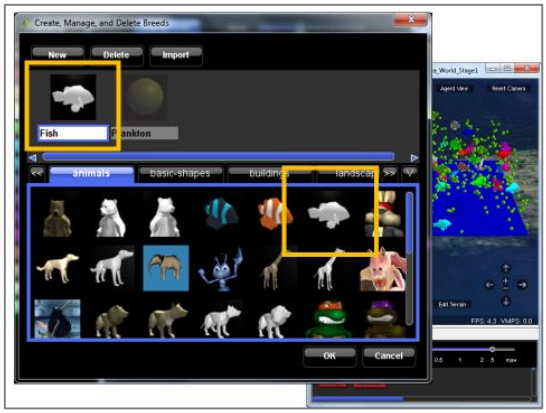
If SpaceLand isn't empty, temporarily disconnect the actions in the Setup procedure, then run it so it clears the world as shown.



Zoom in a little and you'll notice the sea bed is uneven. If you select the Edit Terrain option, the relief is clearer to see. You can alter the terrain by selecting a range of patches and pulling them up or pushing them down, using the options available.



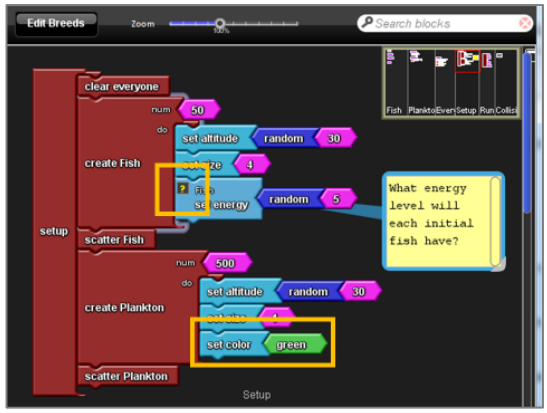
Reconnect your Setup procedure if needed and run it. Our marine world is populated with fish and plankton. These are the two 'breeds' on our world. Select Edit Breeds to investigate them further. Our two breeds are shown right, with the Fish breed highlighted.



Later, if we wish to add another agent, we would do so here. There is a wide range of sprites we can use.

Note that the Fish object that we have chosen to use has no 'skin'. Skinless objects are very useful as their colour can be set in our model. This is how we create different species of fish.

Let's investigate the Setup procedure. Most of the commands here will be familiar. It could be used as an exercise in reading code. Three small points to note.

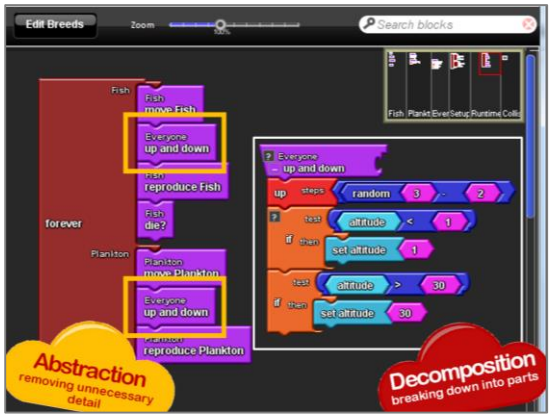


What does 'Set altitude' do? Hovering over the block gives a description. Each fish is set at a random altitude between 1 and 30, above the terrain. You'll find altitude in the 'Traits' drawer. We'll use it to make the fish movement more realistic when we run the simulation.

Note that we set the colour trait of the Plankton to green. But we don't set a colour for the Fish. When a skinless object is created, it is given a random colour. This creates the impression of different species in our simulation.

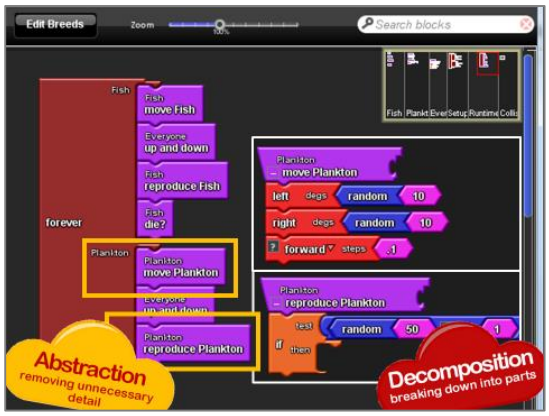
The small '?' indicates a comment. You can add comments to any block in your code, either as explanations or prompts for class discussion. This is a particularly useful feature if you wish to set further exercises along the lines of this one, using partially completed code as scaffolding.

Notice again, how our 'forever' procedure is easy to comprehend because of our procedural abstraction. Activate the 'forever' button in SpaceLand. What happens? The fish and plankton bob up and down but nothing else. Let's investigate the 'up and down' procedure before we explore why the others do nothing. The Up and Down procedure is in the ''Everyone' panel. This means it can be called by any breed. Can anyone decompose the procedure and explain how it works? Try to focus on the precision of the answer students give. It isn't easy to articulate, and working at precision aids students understanding and their ability to express their own code.

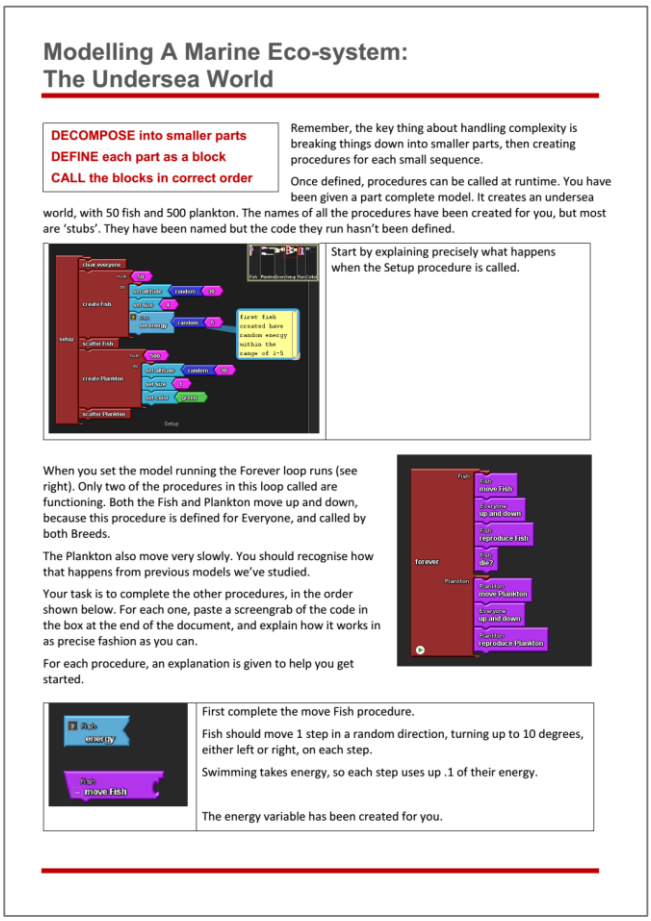


The number of up steps is determined by a calculation. The calculation subtracts 2 from a random number between 1 and 3. This will give a value of either -1, 0 or 1. IF the altitude of the agent is less than 1, the altitude is set to 1. IF it is greater than 30, the altitude is set to 30. If neither condition is met, the agent is moved up or down to the new altitude. The effect is to keep all agents within the altitude bounds of 1 and 30. In the example shown, there are comments to explain this, but the same file has been provided without comments for use with students.

Plankton behaviour is governed by two further procedures. If you observed closely when you ran the simulation you would notice that the Plankton are actually moving. The move Plankton procedure should be familiar – it's our wiggle walk, but the steps are tiny. The reproduce procedure has been left incomplete – an easy first step for pupils to ensure they can make changes and test their model.



The three remaining Fish procedures and the actions required on a collision are left as the initial challenge for the students. An activity sheet, which gives a summary of the actions required for each procedure is included.



Modelling A Marine Eco-system:
The Undersea World

DECOMPOSE into smaller parts
DEFINE each part as a block
CALL the blocks in correct order

Remember, the key thing about handling complexity is breaking things down into smaller parts, then creating procedures for each small sequence.

Once defined, procedures can be called at runtime. You have been given a part complete model. It creates an undersea world, with 50 fish and 500 plankton. The names of all the procedures have been created for you, but most are 'stubs'. They have been named but the code they run hasn't been defined.

Start by explaining precisely what happens when the Setup procedure is called.

When you set the model running the Forever loop runs (see right). Only two of the procedures in this loop called are functioning. Both the Fish and Plankton move up and down, because this procedure is defined for Everyone, and called by both Breeds.

The Plankton also move very slowly. You should recognise how that happens from previous models we've studied.

Your task is to complete the other procedures, in the order shown below. For each one, paste a screengrab of the code in the box at the end of the document, and explain how it works in as precise fashion as you can.

For each procedure, an explanation is given to help you get started.

First complete the move Fish procedure.

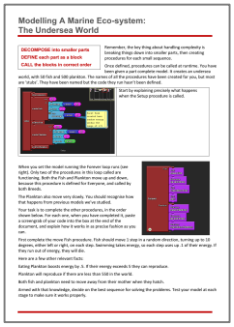
Fish should move 1 step in a random direction, turning up to 10 degrees, either left or right, on each step.

Swimming takes energy, so each step uses up .1 of their energy.

The energy variable has been created for you.

Extending The Model

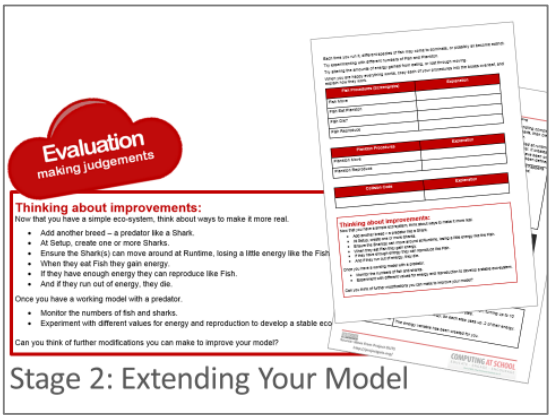
More able students should be able to manage the tasks in the activity sheet without the prompts (shown left), so a slimmed down version is also included (below).



Both ask for students to paste screengrabs of the procedures they develop and explain how they work so the teacher can assess progress.

A completed working version of the model is also included in the resources for reference but other solutions are possible.

Stage 2 involves students extending the model. Like previous exercises there are lots of possibilities. Teachers can exercise judgement about how best to proceed. The suggestions on the activity sheet provide a structured extension. All the required skills have been covered in this session.



However, this could also be an ideal time for students to evaluate their own model and identify other ways to improve it or possibly embark on their own project.

Constructionism, as a theory of learning, holds that children's knowledge develops when building their own creations. StarLogo provides an environment that makes such experimentation possible.

In Conclusion

Nicholas Negroponte once described creating computer programs in these terms: “Children need to learn learning, which is primarily acquired through the passion that comes from access, the ability to make things, to communicate and to express. Writing a computer program, while seemingly esoteric, is in fact the closest a child can come to thinking about thinking. Likewise, debugging a program is the closest one can come to learning learning.” Creating models, such as the ones we have looked at give children a chance to build a genuine ‘computational abstraction’. To what extent do you think there is a more generic educational value?

The scientific importance of emergent behaviour is well summarised in a 12 minute video, taken from the popular American channel NOVA Science Now.

First aired in 2012, it is worth showing, despite the low resolution: youtu.be/Nu7Zcj8HGUk. It makes explicit the interconnected nature of many of the ideas we have touched on and is particularly pertinent as an introduction to Artificial Intelligence.

The full documentary (25 mins) is currently available at youtu.be/qPP9XemRzkA.

In summary, complex adaptive systems are widespread in the natural sciences. There are many examples in political, economic and other social sciences too. They all share the same basic ideas. So modelling techniques can transfer from one context to another. Dynamic, decentralised systems are unpredictable, motivational and easy to model. They are ideal for Key Stage 3. They are an obvious arena in which to develop ‘computational abstractions’, to engage in the wider problem solving involved in building something tangible.

We can summarise by saying these 3 things:

- Computing is about much more than programming.
- Computing is about a way of thinking.
- Computing concepts are transferable to other problem solving contexts.