Chapter 4. Word Play

Glen Bull and Paula Cochran

Alan Turing was an early computing pioneer who worked in the field of artificial intelligence. Turing developed the Turing Test, which proposed that a machine could be described as intelligent if a person communicating with it could not tell if they were speaking with a machine or another person. The first computing languages such as FORTRAN were designed for numeric operations, but sentences used in conversations consist of lists of words rather than numbers. Another computing pioneer working in the Artificial Intelligence Laboratory at M.I.T., John McCarthy, developed a programming language named *LISP* (short for "list processing" language) that was well suited for development of programs designed to address the challenge posed by Turing.

Seymour Papert, who subsequently served as co-director of the M.I.T. Artificial Intelligence Laboratory, developed a dialect of LISP known as *Logo*. (*Logo* is the Greek word for "word.") *Logo* was designed for use by children, and included many features well-suited for exploration of the lists of words and sentences that comprise a language. The descendants of *Logo* such as *Scratch* and *Snap!* retained these list processing capabilities.

The Grammatical Structure of Gossip

Just as a computer program has a structure, language also has a structure. Sentences, poems, and stories all have an underlying composition. The way in which sentences are put together is described as their *syntax*. The structure of the underlying meaning is described as *semantics*. The way in which syntax and semantics are used to form a story can be described as *style*.

In *Exploring Language with Logo*, Paul Goldenberg investigates these concepts through development of a grammar of gossip. In its simplist terms, gossip can be described as:

[Who] [Does What]

Who can be constructed from a list of names:

Sam Sally George Ann

Does What can be constructed from a list of actions.

[procrastinates] [talks all the time] [cheats] [loves to gossip]

In *Snap!* these lists can be constructed in the following way:

Making a List

The *Snap!* list command is found under the *Variables* section of the commands palette. In this illustration, a list of names has been created.

A second list includes a series of actions.

If, for example, a variable is created and named "people", the **set** command block can be used to allow your program to use any item from the first list as a value for that variable.

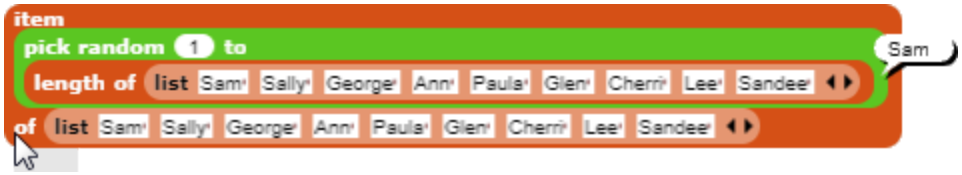
In the *Gossip* game, a name will be randomly picked from the list.

Picking a Random Item from a List

The *Random* function (found under the *Operators* command palette) can randomly choose a number between 1 and some upper bound. Since there are nine words in the list of people, in this case we would like to pick an item between one and nine.

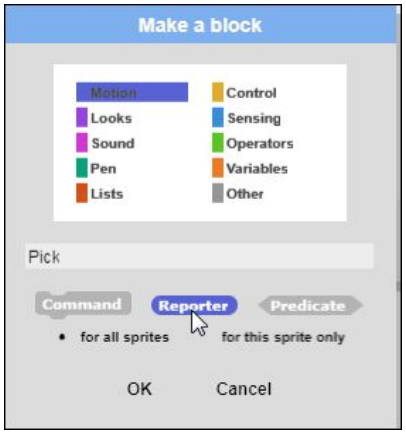
At times there may be a different number of items in the list. The *Length of List* command (found under the *Variables* command palette) can be used to authomatically find the length of any list. In this case, the *Length of List* command tells us that there are nine items in the list of people

The Random command now picks a number between 1 and the number of items in the list.



The Pick Procedure

These commands enable us to create a more general procedure named *Pick* that randomly picks an item from a list and reports the result. Since the *Pick* procedure reports an item randomly selected from a list the *Reporter* option is chosen when the procedure is created.



The completed procedure looks like this.

The *Pick* procedure randomly picks an item from a list.

The Pick procedure can now be used to make a procedure that randomly picks a person from a list. The procedure, named Who, reports the name of the person picked from the list. Because the procedure reports an outcome, the oval *Reporter* option is chosen when the procedure is created.

The completed Who procedure reports the name of the person picked from the list.

A parallel *Does What* procedure picks an action from a list

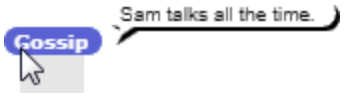
Gossip

The *Join* operator (found under the *Operators* section of the command palette) can be used to join a person and action together in a sentence.

A space can be added to separate the person and the action and a period can be added at the end to complete the sentence.

These commands can be used as the basis of the *Gossip* procedure.

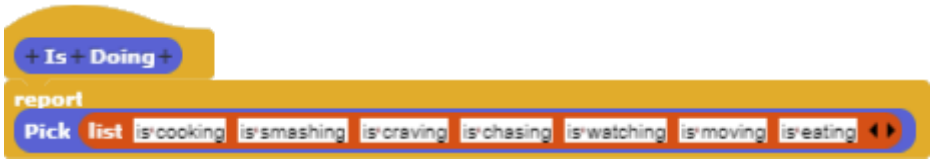
The *Gossip* program reports a person and an action.



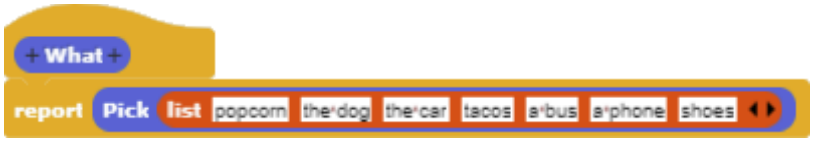
This format can serve as a springboard for exploration of more complex grammars. For example, *Does What* could be replaced with a transitive verb, *Is Doing*.

[Who] [Is Doing] [What]

The *Is Doing* procedure includes a list of actions that people are doing.



This can be combined with a word picked from the following list.



This leads to construction of sentences like this.