Unit 2: Programming

Dave Abel

February 12th, 2016
Outline for Today

› Revisit clicker question from Wednesday

› Finish Scratch Overview (Lists, Randomness)

› Demystify *how* programming can reconfigure a computer

› Close the unit with a quick look at other languages
Clicker Question!

Q: Which loop will move the cat more steps?
Q: Which loop will move the cat more steps?
Clicker Question!

Why not [C]
Clicker Question!

Forever:
1) if counter is 10, move
2) add 1 to counter

Why not [C]
Clicker Question!

Forever:
1) if counter is 10, move
2) add 1 to counter

Counter will only be set to 10 one time!

Why not [C]
Things You’ll Do in Scratch

‣ **Machine Learning:** Writer a classifier, similar to how your email determines what is “Spam” and what is “Not Spam”!

‣ **Programming:** A simple game

‣ **Vision + NLP:** Model Roald Dahl’s style of writing!

‣ **Recursion:** Draw pretty pictures

‣ And more…
Things You’ll Do in Scratch

- **Machine Learning**: Write a classifier, similar to how your email determines what is “Spam” and what is “Not Spam”!

- **Programming**: A simple game

- **Vision + NLP**: Model Roald Dahl’s style of writing!

- **Recursion**: Draw pretty pictures

- And more…
Block Types

Event/Trigger (e.g. when clicked)
Statement (e.g. move cat)
Ending Statement
Boolean Value
Numeric Value
String Value
Container (loops, if statements)
More Scratch Things: Lists

More Scratch Things: Lists


Q: Is there a kiwi in the “basket”?

Q: What’s the biggest number in “numbers”?
Lists

Index

1. Persimmon
2. Apple
3. Apricot
4. Avocado
5. Banana
6. Bilberry
7. Blackberry
8. Blackcurrant
9. Blueberry
10. Boysenberry
11. Cantaloupe
Lists

Index

Contents

1. Persimmon
2. Apple
3. Apricot
4. Avocado
5. Banana
6. Bilberry
7. Blackberry
8. Blackcurrant
9. Blueberry
10. Boysenberry
11. Cantaloupe
Lists

Let's take a look!

Index

Contents

1. Persimmon
2. Apple
3. Apricot
4. Avocado
5. Banana
6. Bilberry
7. Blackberry
8. Blackcurrant
9. Blueberry
10. Boysenberry
11. Cantaloupe
Clicker Question:

Q: What will “numbers” look like after running this block?
Clicker Question:

Q: What will “numbers” look like after running this block?

[A]  
```
numbers
1 2
2 20
3 17.4
4 3
5 30
6 47
7 -4.2
```

[B]  
```
numbers
1 2
2 20
3 thing
4 3
5 3
6 30
7 47
```

[C]  
```
numbers
1 2
2 20
3 17.4
4 thing
5 3
6 30
7 47
```
Clicker Question:

Q: What will “numbers” look like after running this block?

Hint: Think about the basket!
Clicker Question:

Q: What will “numbers” look like after running this block?

 replace item 3 of numbers with thing

Hint: Think about the basket!
Clicker Question:

Q: What will “numbers” look like after running this block?

Hint: Think about the basket!
Clicker Question:

Q: What will “numbers” look like after running this block?

Hint: Think about the basket!

We will always grab thing from lists by their index.
Coin Flipping!

pick random 1 to 10

And more… Let’s take a look!
Clicker Question!

Q: What does the mystery block do?
Q: What does the mystery block do?

(A) Counts the number of total items in the list “basket”

(B) Finds the longest fruit name in “basket”

(C) Looks through basket, counting the number of items with the name parameter

(D) Finds the index of the item with the name of parameter

(E) Looks through basket to determine if an item with the name parameter is in it.
Q: What does the mystery block do?

(A) Counts the number of total items in the list “basket”

(B) Finds the longest fruit name in “basket”

(C) **Looks through basket, counting the number of items with the name parameter**

(D) Finds the the index of the item with the name of parameter

(E) Looks through basket to determine if an item with the name parameter is in it.
Recall: Unit 2 Takeaway

Programming lets us reconfigure what a computer does!
Recall: Unit 2 Takeaway

Programming lets us reconfigure what a computer does!

Seen some examples:
Recall: Unit 2 Takeaway

Programming lets us reconfigure what a computer does!

So how does this work?
The Computer

- Internal states
- Complex logic relating bits
- Mechanism for setting bits (input)
- Mechanism for displaying bits (output)
The Computer

- Internal states
- Complex logic relating bits
- Mechanism for setting bits (input)
- Mechanism for displaying bits (output)
The Computer

Memory
The Computer

Memory

100010101010101001010100011111100101001010010100
The Computer

Memory

```
100010101010010101000111111101010010100 ....
```

```
1001...
0011...
0101...
1110...
0001...
1001...
```

32 bits 32 bits 32 bits 32 bits 32 bits 32 bits

32 bit operating system
The Computer

Memory

64 bits

64 bit operating system
The Computer

Memory

1000101010100101010001111111001010010100 

1001... 0011... 0101... 1110... 0001... 1001...

10KB = 10 Kilobytes
The Computer

Memory

100010101010010101000111111001010010100

1001...

0011...

0101...

1110...

0001...

1001...

10KB = 10 Kilobytes

1 Kilobyte = 1024 bytes
The Computer

Memory

100010101010010101000111111001010010100 ....

1001 .... 0011 .... 0101 .... 1110 .... 0001 .... 1001 ....

10KB = 10 Kilobytes

1 Kilobyte = 1024 bytes
# A Quick Tangent

<table>
<thead>
<tr>
<th>Number of Bytes</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2^{10} = 1,024$</td>
<td>1 Kilobyte</td>
</tr>
<tr>
<td>$2^{20} = 1,048,576$</td>
<td>1 Megabyte</td>
</tr>
<tr>
<td>$2^{30} = 1,073,741,824$</td>
<td>1 Gigabyte</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$2^{100} = 1,267,650,600,228,229,401,496,703,205,376$</td>
<td>???</td>
</tr>
</tbody>
</table>

Approximate # atoms in the known universe: $10^{80}$
The Computer

Memory

100010101010010101000111111001010010100

.....

1001... 0011... 0101... 1110... 0001... 1001...

10KB = 10 Kilobytes = 10 * 1024 bytes

1 Kilobyte = 1024 bytes
The Computer

Memory

100010101010010101000111111001010010100  

1001  
0011  
0101  
1110  
0001  
1001  

= 10 * 1024 bytes
The Computer

Memory

100010101010010101000111111001010010100001001...

= 1001... 0011... 0101... 1110... 0001... 1001...
The Computer

Memory

100010101010010101000111111001010010100

.....
The Computer

Memory

100010101010010101000111111001010010100

.....

1001...
0011...
0101...
1110...
0001...
1001...

change  color  effect  by  25
The Computer

Memory

100010101010010101000111111001010010100

...
The Computer

Commands themselves are also binary
The Computer
The Computer
The Computer

CPU

1101…
The Computer

CPU

gets translated into Machine Code:

```
1000 add $a0,$s0,$zero  # $a0 = x
1004 add $a1,$s1,$zero  # $a1 = y
1008 addi $ra,$zero,1016  # $ra=1016
1012 j  sum  # jump to sum
1016 ...
2000 sum: add $v0,$a0,$a1
2004 jr $ra  # jump to 1016
```
The Computer

Memory

100010101010010101000111111001010010100

1001...

0011...

1101...

100010101010010101000111111001010010100

CPU
The Computer

Memory

100010101010010101000111111001010010100

...

1001...
0011...
...
1101...

CPU
The Computer

Memory

1000101010100101010001111111001010010100

.....

1001...
0011...
...
1101...

CPU
The Computer

Memory

100010101010010101000111111001010010100

.....

CPU
Recap!

Becomes *machine code* that carries out the “When flag clicked” block, in binary and logic.
Recap!

Becomes *machine code* that carries out that block in binary and logic.
Abstraction
Abstraction

```
1000 add $a0,$a0,$zero  # $a0 = x
1004 add $a1,$a1,$zero  # $a1 = y
1008 addi $ra,$zero,1016 # $ra=1016
1012 j sum             # jump to sum
1016 ...               
2000 sum: add $v0,$a0,$a1
2004 jr $ra             # jump to 1016
```
Abstraction

x, y, size

when clicked
  go to x: 0 y: 0
  clear
  pen down
  repeat 3
    move 80 steps
    turn 120 degrees
  wait 1 secs
  point in direction 90°

1000 add $a0, $s0, $zero # $a0 = x
1004 add $a1, $s1, $zero # $a1 = y
1008 addi $ra, $zero, 1016 # $ra=1016
1012 j sum # jump to sum
1016 ...
2000 sum: add $v0, $a0, $a1
2004 jr $ra # jump to 1016
Abstraction

```
1000 add $a0,$s0,$zero    # $a0 = x
1004 add $a1,$s1,$zero    # $a1 = y
1008 addi $ra,$zero,1016  # $ra=1016
1012 j  sum               # jump to sum
1016 ...
2000 sum: add $v0,$a0,$a1
2004 jr $ra                # jump to 1016
```
Abstraction

when clicked
  go to x: 0 y: 0
clear
ten down
repeat 3
  move 80 steps
turn 120 degrees
  wait 1 secs

point in direction 90°

1000 add $a0,$s0,$zero   # $a0 = x
1004 add $a1,$s1,$zero   # $a1 = y
1008 addi $ra,$zero,1016  # $ra=1016
1012 j sum     # jump to sum
1016 ...
2000 sum: add $v0,$a0,$a1
2004 jr $ra         # jump to 1016
“Natural Language Programming”

"Cat, draw a triangle"

x, y, size

triangle
“Natural Language Programming”

“Cat, draw a triangle”

Problem:

- 1 triangle
- 2 triangles
- 3 triangles
Other Languages

Python

```
counter = 0
while not(counter == 10):
    print counter
    counter = counter + 1
```

Java

```
public static void main(String[] args) {
    int counter = 0;
    for(counter = 0; counter != 10; counter++) {
        System.out.println(counter);
    }
}
```
Other Languages: Prolog

\begin{verbatim}
mother_child(trude, sally).

father_child(tom, sally).
father_child(tom, erica).
father_child(mike, tom).

sibling(X, Y) :- parent_child(Z, X), parent_child(Z, Y).

parent_child(X, Y) :- father_child(X, Y).
parent_child(X, Y) :- mother_child(X, Y).
\end{verbatim}
Esoteric Languages: CHEF

*Lobsters with Fruit and Nuts.*

**Ingredients.**

- 72 g hazelnuts
- 101 eggs
- 108 g lobsters
- ...

**Method.**

- Put lemon juice into the mixing bowl.
- Put passion fruit into the mixing bowl.
- Put durian into the mixing bowl.
- Put lobsters into the mixing bowl.
- ...

...
Esoteric Languages: Shakespeare

Romeo, a young man with a remarkable patience.

Juliet, a likewise young woman of remarkable grace.

Ophelia, a remarkable woman much in dispute with Hamlet.

Hamlet, the flatterer of Andersen Insulting A/S.

…

Act I: Hamlet's insults and flattery.

Scene I: The insulting of Romeo.

[Enter Hamlet and Romeo]

Hamlet:

You lying stupid fatherless big smelly half-witted coward! You are as stupid as the difference between a handsome rich brave hero and thyself!

Speak your mind!

…
Unit 2 Reflection

‣ Scratch!
  • Drawing Squares
  • Loops, Conditions
  • Making Blocks
  • Variables
  • Lists
  • Randomness

‣ How do programs reconfigure a computer?
Unit 2: Takeaway

1. Physical gates are inflexible.

2. Programming lets us reconfigure what a computer does!
Up Next: Algorithms!

- We’ve seen that programs can solve certain problems, e.g. “What’s the smallest number in my list?”

- What other problems are solvable with programs?

- What ones are easy? What ones are hard?

- The study of algorithms is the *formal study of solving problems using computational tools!*

- **Central idea of computer science.**