

# Voice Programming in Computer Science Education

February 6, 2018

## 1 Introduction

- 1.1 What is the goal of the project? What problem does it solve?
- 1.2 Who is meant to benefit?
- 1.3 What are the results or outcome of the project?

## 2 HCI Principles

- 2.1 What HCI principles were kept in mind in the design of our system?
- 2.2 How did those HCI principles influence our design decisions?

## 3 Blockly

- 3.1 What is the Blockly environment?
- 3.2 What is the goal of the Blockly project?
- 3.3 How are blocks arranged horizontally? What language do we use to describe this?
- 3.4 How are blocks arranged vertically? What language do we use to describe this?
- 3.5 Is just one chain of blocks allowed or several?
- 3.6 If several, how are they executed?
- 3.7 How does the user typically interact with Blockly?
  - 3.7.1 What's the medium (hint: mouse)?
  - 3.7.2 What actions are available?
  - 3.7.3 How can we group them into categories?
  - 3.7.4 How does moving blocks work?
  - 3.7.5 How does deleting blocks work?
  - 3.7.6 How does connecting blocks work?
  - 3.7.7 How does separating blocks work?

## 4 Turtle

- 4.1 What is the turtle game?
- 4.2 What set of blocks are available to complete the game?
- 4.3 What do these blocks do?
- 4.4 Why is the game divided into levels?
- 4.5 How do the levels serve to teach different programming concepts?

## 9 Layout

As the user writes a program, certain locations on the workspace fill with blocks. This presents a challenge: what should happen when blocks overlap with one another? For our system to be practically useful, it must avoid introducing such visual impairments, which may prevent the user from issuing commands (e.g. if they cannot see a block ID) or which may introduce unnecessary cognitive load.

Blockly by default places new blocks at the top left of the workspace, even if the new block will overlap an old block. Similarly, when Blockly connects one block to another, the resulting chain might overlap another chain.

We devise simple layout algorithms for each case. To make this formal, we view the workspace as a 2D plane.

**Adding Blocks** (1) Find the vertically lowest block on the workspace. Let  $y$  be the vertical coordinate of its lower left corner. (2) Place the new block such that its top left corner is at  $(20, y)$ , that is, below the lowest block on the workspace, 20 pixels right of the flyout.

**Moving Blocks** Suppose the user has just connected one block to another, and the resulting chain now overlaps  $n$  blocks. We repair the layout by moving the  $n$  conflicting blocks via the procedure below. For brevity, let  $W$  and  $H$  be the width and height of the entire workspace, respectively, let  $b$  be the block to be moved, and let  $w_b$  and  $h_b$  be the width and height of  $b$  and its children. Additionally define  $m$  to be a small margin.

1. Construct the set of candidate new locations:

$$C = \{(x, y) = (rm, sm) : r, s \in \mathbb{Z}, x \leq W, y \leq H\}$$

2. For every  $(x, y) \in C$  check if there is no block within the box bounded by the four points:

$$(x, y), (x + w_b, y), (x, y + h_b), (x + w_b, y + h_b)$$

Let  $(\hat{x}, \hat{y})$  be the first found satisfying that condition. If none is found, set  $\hat{x} = \hat{y} = m$ .

3. Move  $b$  so that its top left corner is located at  $(\hat{x}, \hat{y})$ .

We choose  $m = 20$  (pixels) because it is small enough to preserve space on the workspace, and large enough to prevent Blockly from automatically connecting nearby blocks.

## **10 Related Work**

- 10.1** What is the related work in speech interfaces for programming?
- 10.2** What is the related work in speech interfaces for computer science education?

## **11 Future Work**