Formative SAGE Assessments: Parsons’ Programming

Midterm Progress Report

Akanksha Gupta

Sudhanshu Mohan

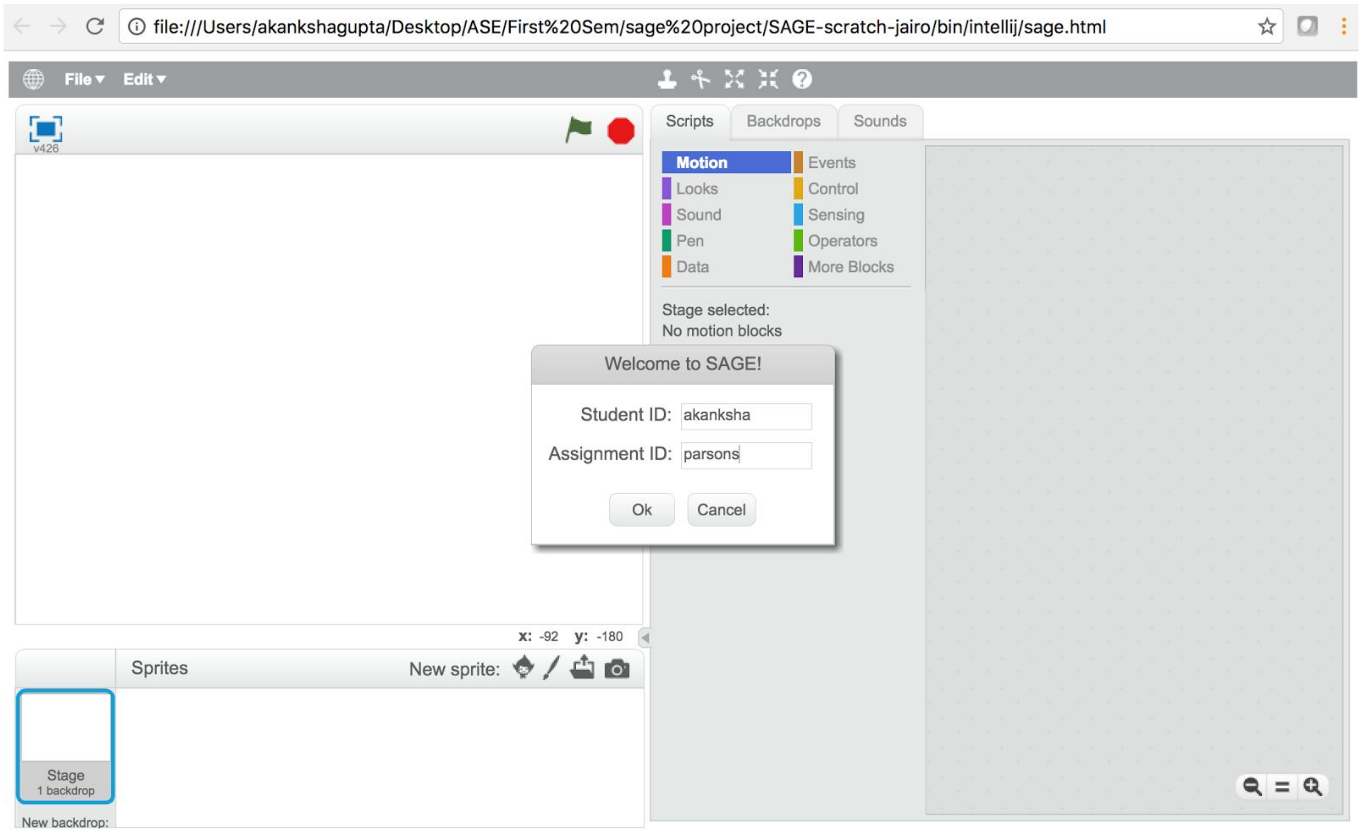
COMS E6901, Section 14 Fall 2016

1.0 Introduction

This document will describe the technical work completed up to this point for the SAGE Assessment-Parsons' Programming project. Section two of this document will describe the design and implementation of the parson's concept in SAGE. Section three will describe the assessment server that is used to parse Scratch project and assessment files to evaluate student SAGE submissions. Section four will describe future work.

2.0 Earlier SAGE

The earlier SAGE allowed the teachers to create assignments for students and involve students in a computational learning experience. The earlier SAGE screenshot is as follows.



Stage selected:
No motion blocks

Welcome to SAGE!

Student ID: akanksha

Assignment ID: parsons

Ok Cancel

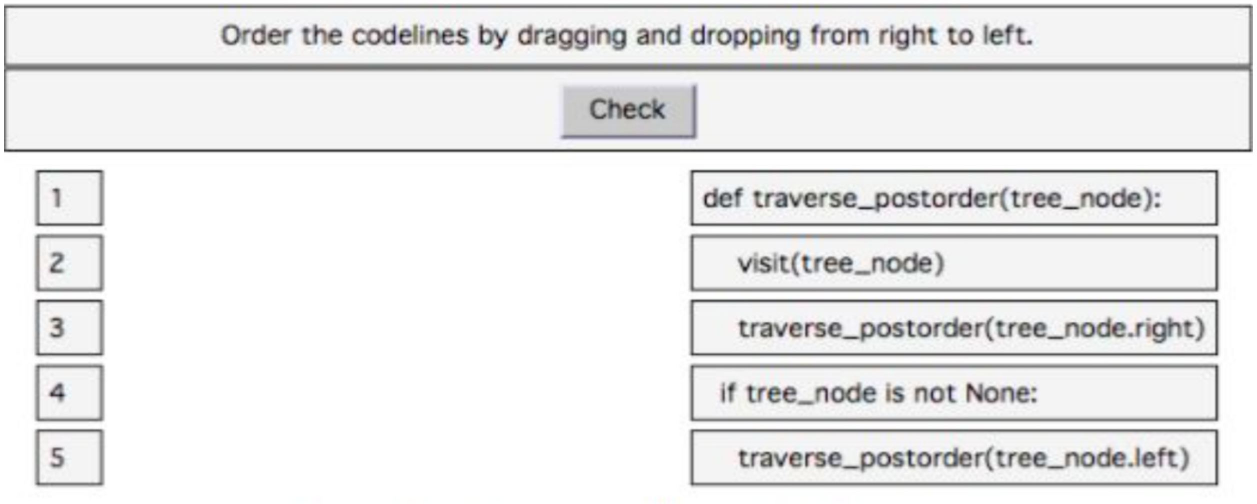
3.0 Concept of Parson

Parson’s programming puzzles are a fun and interactive way that allows teachers/tutors to touch upon important topics and syntactic and logical errors in a program. Such puzzles expose students to short pieces of code that help students construct the logical order of statements for any Parson coding puzzle. *S*ince each puzzle solution is a complete sample of well-written code, use of the tool exposes students to good programming practices. Parson’s puzzles are built on the idea of a drag and drop concept where a student puts the block with the code piece in logical order by dragging and dropping at the correct order. A family of code is given where lines of code are provided and the goal of the students is to reach the final correct solution by sorting and possibly selecting the correct code lines. This reduces the cognitive load for the student as the code is already present.

4.0 Related Work

4.1 Hot Potatoes

The original Parson’s problems (Parsons & Haden, 2006) were created using a generic drag-and- drop exercise framework called Hot Potatoes . Exercises created with this tool can be exported to HTML and JavaScript pages. Exercises are solved by dragging lines from right to left (shown in Figure 2). When feedback is requested, lines in (absolutely) correct positions are highlighted. One problem of this UI is that inserting a line between two existing lines is cumbersome. Student may need to move all lines after the insertion point to create a free slot where the new line can be inserted.


Order the codelines by dragging and dropping from right to left.

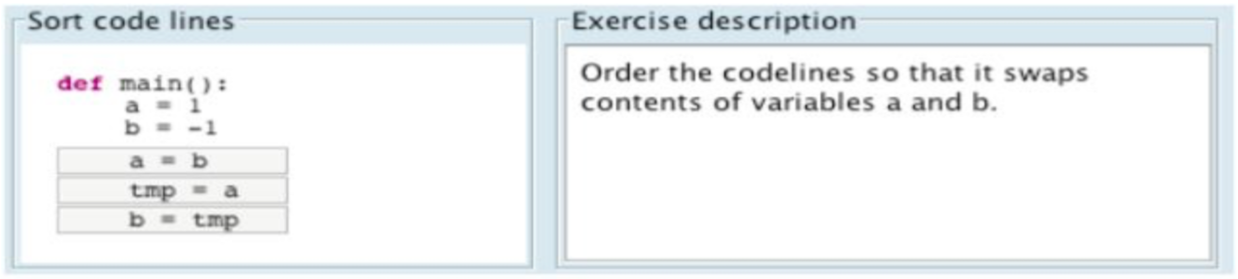
```
def traverse_postorder(tree_node):
   visit(tree_node)
   traverse_postorder(tree_node.right)
 if tree_node is not None:
   traverse_postorder(tree_node.left)
```

Figure 4: A Parson’s problem in Hot Potatoes

Figure 2: A Parson’s problem in Hot Potatoes

4.2 ViLLE

ViLLE (Rajala, Laakso, Kaila, & Salakoski, 2007) is a Java application/applet which allow context to be created around the editable code. Distractors are not supported. The feedback is an error message in case the resulting code does not compile or a number of points in case it does. In this case, the student can also step through a visualization of the execution of his/her solution. An example of the interface is shown in Figure 3.



```
def main():
    a = 1
    b = -1
    a = b
    tmp = a
    b = tmp
```

Order the codelines so that it swaps contents of variables a and b.

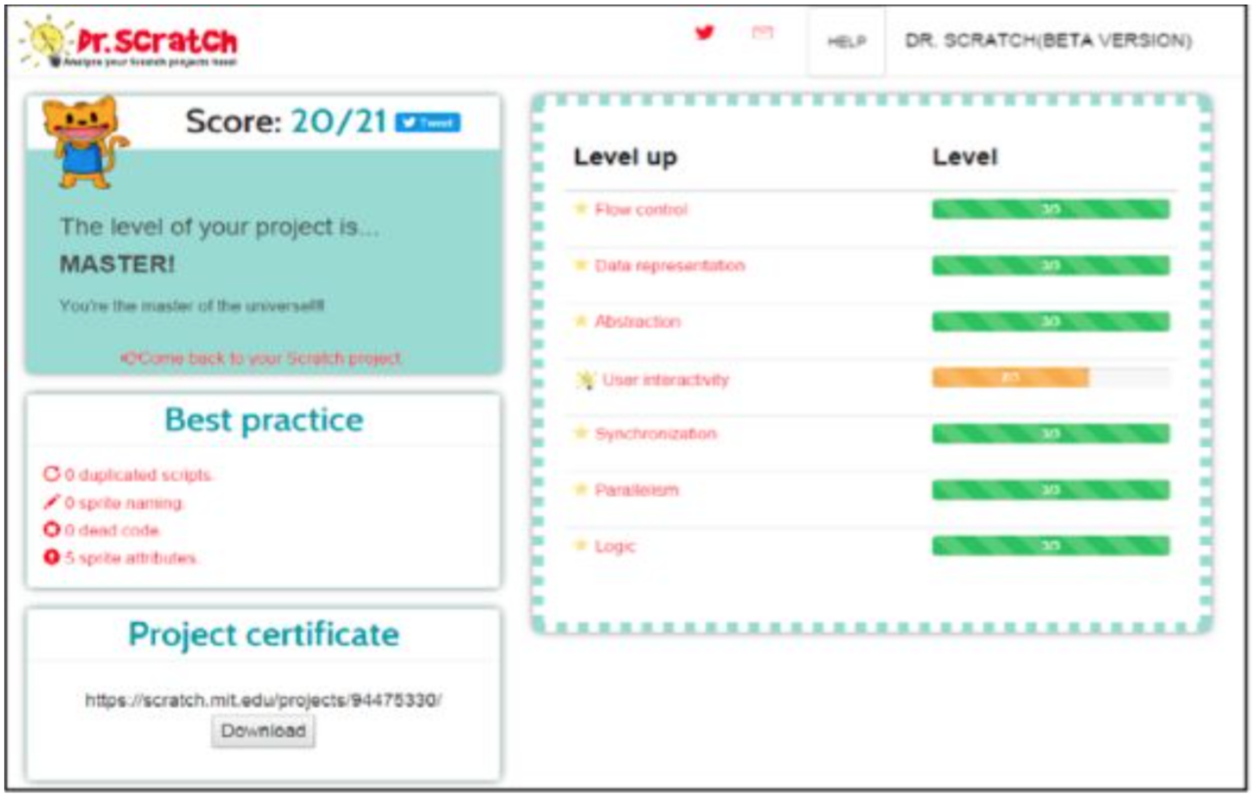
Figure 5: ViLLE
Figure 3: ViLLE

4.3 CORT

CORT (Garner, 2007) has been used with Visual Basic programs so that students move lines from left to a part-complete solution on the right. Moving the lines is done by selecting a line and clicking arrow buttons to move it left or right. To get feedback, student can copy the code into Visual Basic interpreter and execute the code. CORT supports both distractors and context.

4.4 Dr. Scratch

Dr. Scratch (Moreno-Leon, Robles, & Roman-Gonzalez, 2015) is a web application where Scratch project files can be uploaded for automated analysis. It serves as an analytical tool that evaluates Scratch projects in a variety of computational areas. After a project file is analyzed, a scorecard is presented that indicates how well the project uses a variety of computational thinking concepts. An example of this score card is illustrated in Figure 6. Dr. Scratch helps educators in the assessment of student projects and encourages students to improve their programming skills in a fun way. Feedback from Dr. Scratch enables students to understand that successful completion of an assignment includes more than just completing a set of tasks. Successful completion also includes mastering computational thinking concepts that improve their ability to complete similar assignments in the future even if they are not in the same domain.


Dr.Scratch
Analyze your Scratch projects here!
HELP
DR. SCRATCH(BETA VERSION)
Tweet
Download

Score: 20/21

The level of your project is...

MASTER!

You're the master of the universe!!!

Come back to your Scratch project

Best practice

- 0 duplicated scripts.
- 0 sprite naming.
- 0 dead code.
- 5 sprite attributes.

Project certificate

https://scratch.mit.edu/projects/94475330/

Level up	Level
Flow control	3/3
Data representation	3/3
Abstraction	3/3
User interactivity	2/3
Synchronization	3/3
Parallelism	3/3
Logic	3/3

Figure 6: Dr.Scratch scorecard

The proposed project can benefit from Dr. Scratch by taking inspiration from the approach of using a scorecard to present students with visual feedback on the progress they are making on the assignment. The scorecard is concise and clear. Students immediately see where they have mastered a concept and where they need to improve. Badges and scores in Dr. Scratch helps keep students engaged because it introduces an added incentive of social competitiveness among classmates. And the progress bars allow students to quickly understand their progress towards completing their assignment the way it was really meant to be done by their teacher.

5.0 Features Added

Parsons concept now developed in SAGE empowers teachers to play the role of game designers. This enables them to create parsons programming puzzles for students to play as they learn computational thinking concepts. The teacher designs parsons puzzle that will have a combination of sensing, motion and operators and students play the game. To indulge students in learning particular concepts and not be distracted by the various palettes available on SAGE, we have included the feature that allows the teacher to design the complete Parson's palette dynamically.

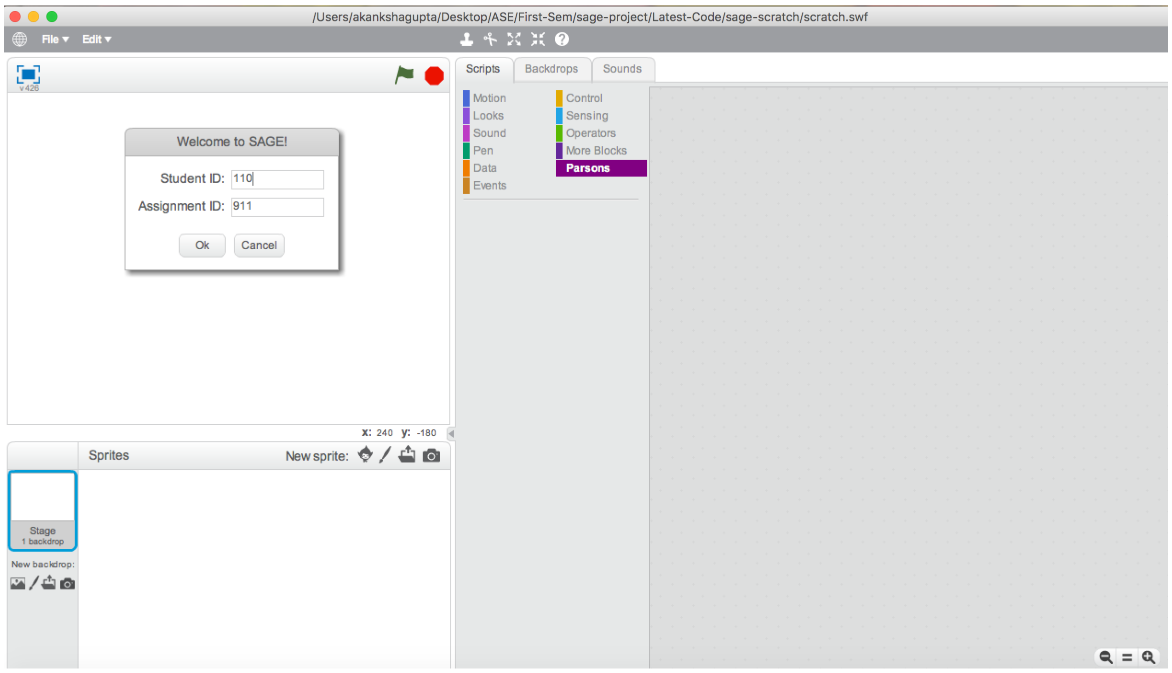
Every block provides a checkbox that can be selected/deselected based on teacher's requirement. If the teacher selects the block, it dynamically gets populated to the Parson's palette. Further to ensure consistency, we have ensured that a block that is excluded but selected by the teacher doesn't get populated into Parson's palette. This is to ensure consistency in designing of puzzles. Only if a block is included and selected by the teacher, the block gets populated in the Parson's palette. After the teacher designs the Parson's palette, the palette is used to design the puzzle.

Also the teacher adds the sprite and sensing blocks for creating an interactive by having sprites. Sprites, either user-created, uploaded, or found in the sprites library, are the objects that perform actions in a project.

6.0 Implementation

The Scratch 2.0 editor is implemented using ActionScript and Flex. The code for the editor was written by the Lifelong Kindergarten Group at the MIT Media Lab and is publically available on GitHub under the GPL v2 license (MIT Lifelong Kindergarten Group, 2016). This Formative SAGE Assessment project extends the fork created by Jeff Bender (Bender, 2015).

This is the dashboard page. Here one can enter the Student Id and Assignment id. One can now see the Parson palette added now to the palette's section.

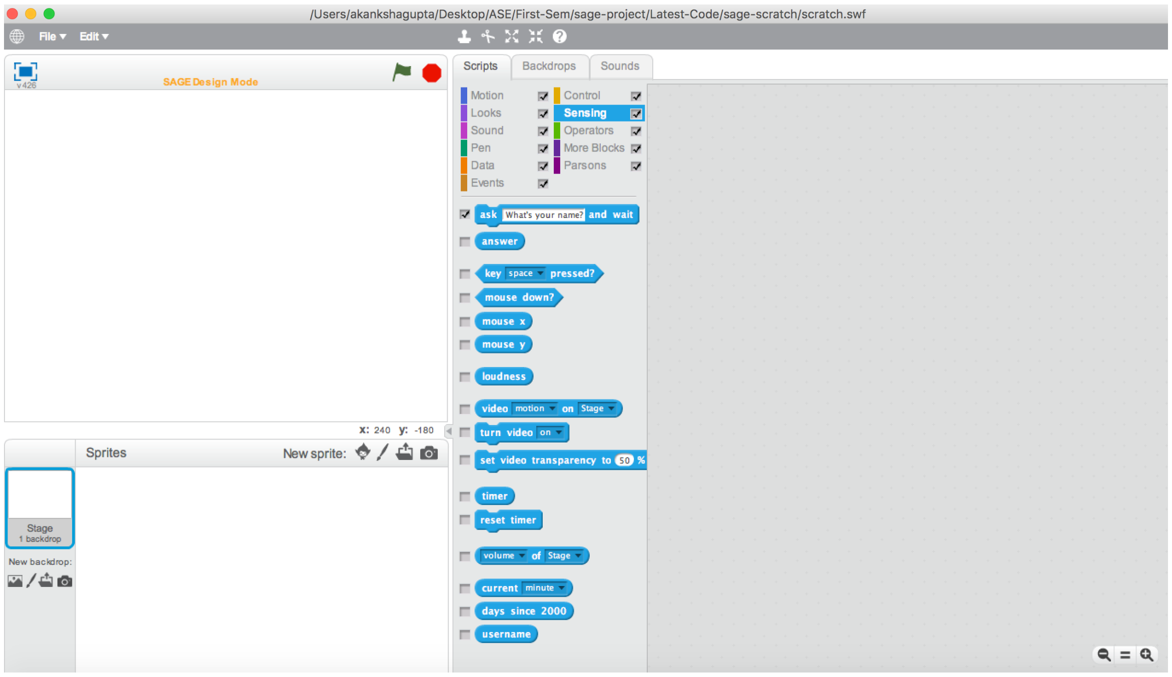

/Users/akankshagupta/Desktop/ASE/First-Sem/sage-project/Latest-Code/sage-scratch/scratch.swf

Welcome to SAGE!

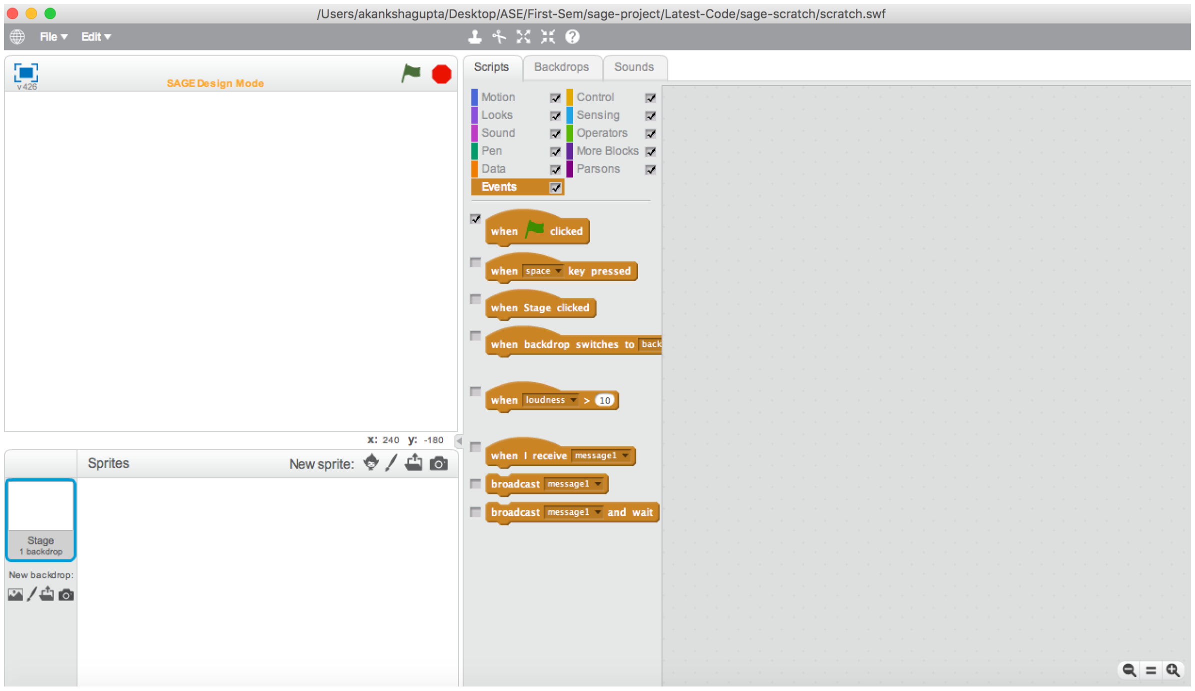
Student ID: 110

Assignment ID: 911

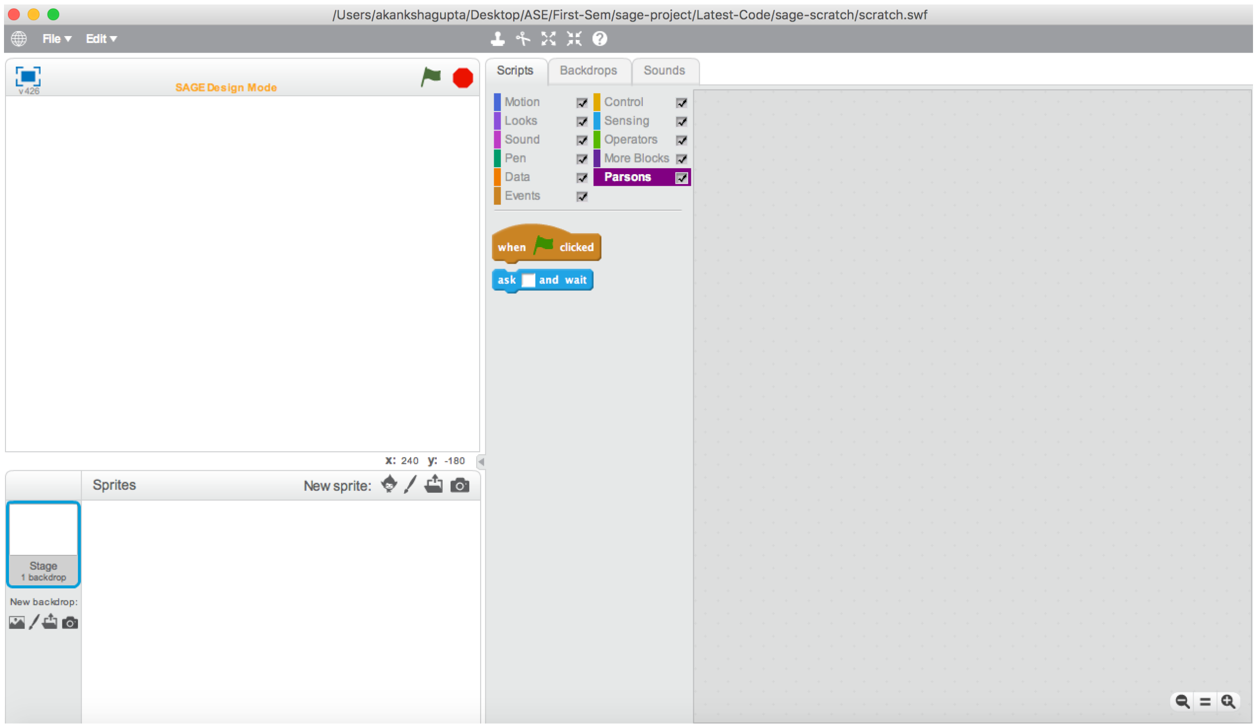
The teacher can start designing for Parson's in the SAGE design mode. When we click initially on Parson's palette, it has no blocks and is blank. Next, we click on any other existing palette. Corresponding to each block, now we have a checkbox on the left. Selection of a block's checkbox auto populates the block in the Parson's palette.



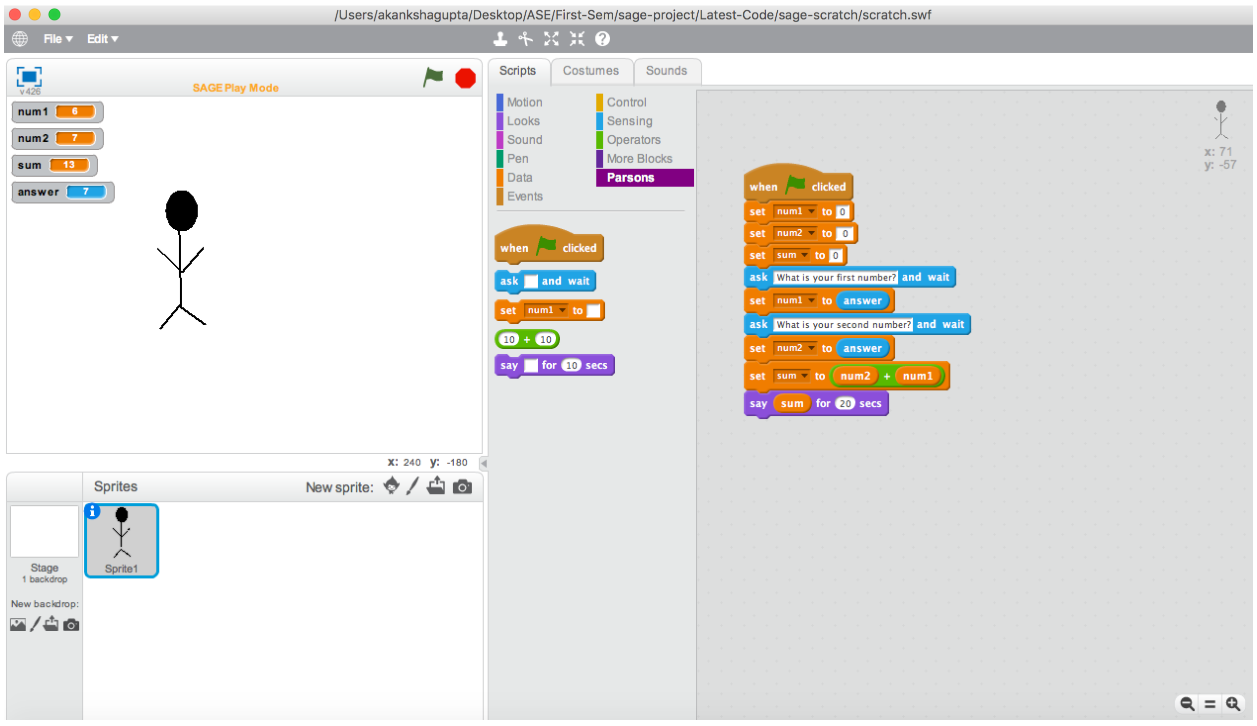
Similarly, we drag and drop blocks from other palettes as well.



Now we see the selected blocks in the Parson's palette from where the teacher can drag and drop and design the Parson's puzzle. This leads to guided learning for the students as they use only the blocks the teacher wants them to actually use.

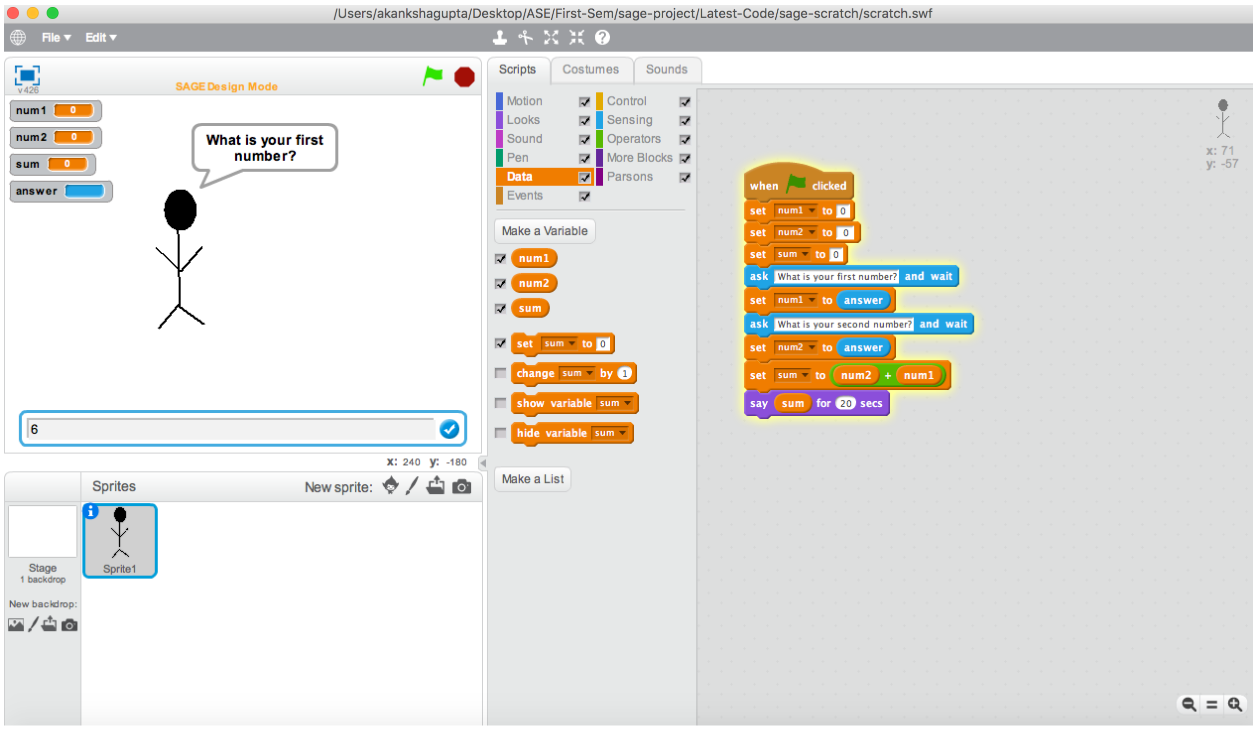


This is a sample parsons' programming puzzle designed in SAGE, teaching the concept of sum of two numbers. Also for playful experience, we design a sprite on the stage. The sprite prompts the user to input two numbers and the displays the sum of the two entered numbers.



```
when clicked
set num1 to 0
set num2 to 0
set sum to 0
ask What is your first number? and wait
set num1 to answer
ask What is your second number? and wait
set num2 to answer
set sum to num2 + num1
say sum for 20 secs
```

First it prompts to enter the first number.


/Users/akankshagupta/Desktop/ASE/First-Sem/sage-project/Latest-Code/sage-scratch/scratch.swf

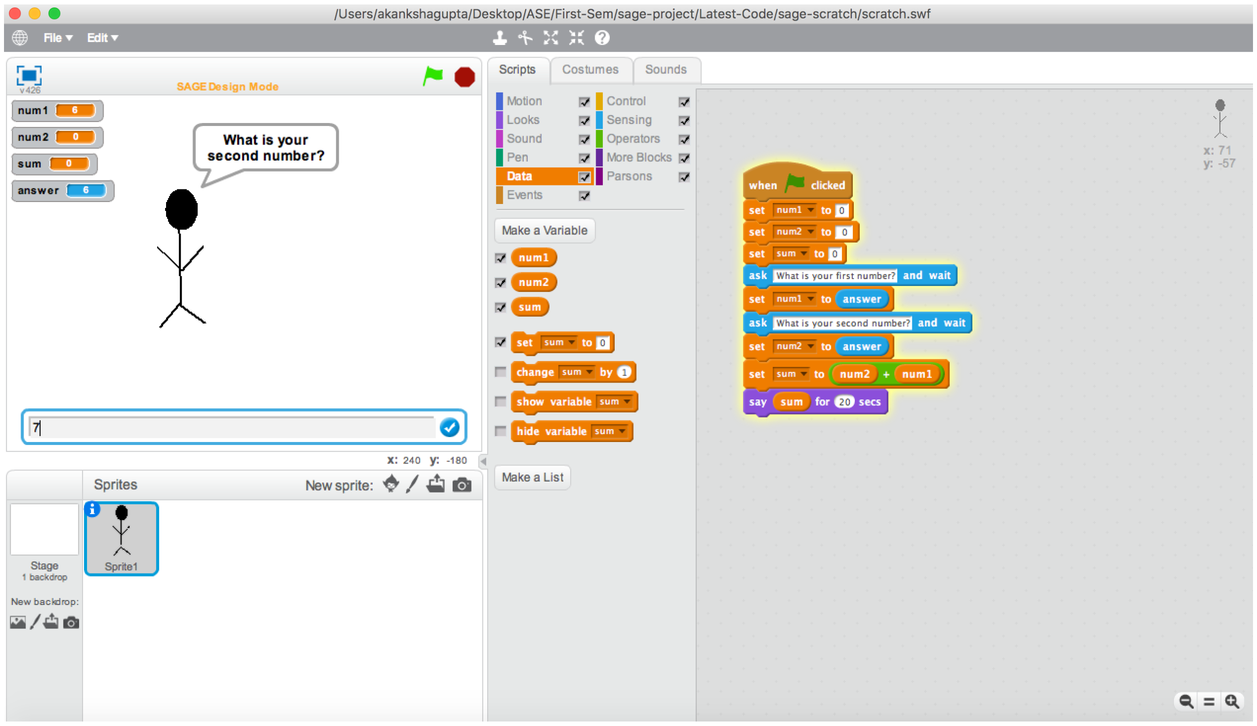
num1 0
num2 0
sum 0
answer

What is your first number?

6

```
when [green flag] clicked
set num1 to 0
set num2 to 0
set sum to 0
ask What is your first number? and wait
set num1 to answer
ask What is your second number? and wait
set num2 to answer
set sum to num2 + num1
say sum for 20 secs
```

Next, it prompts to enter the second number.


/Users/akankshagupta/Desktop/ASE/First-Sem/sage-project/Latest-Code/sage-scratch/scratch.swf

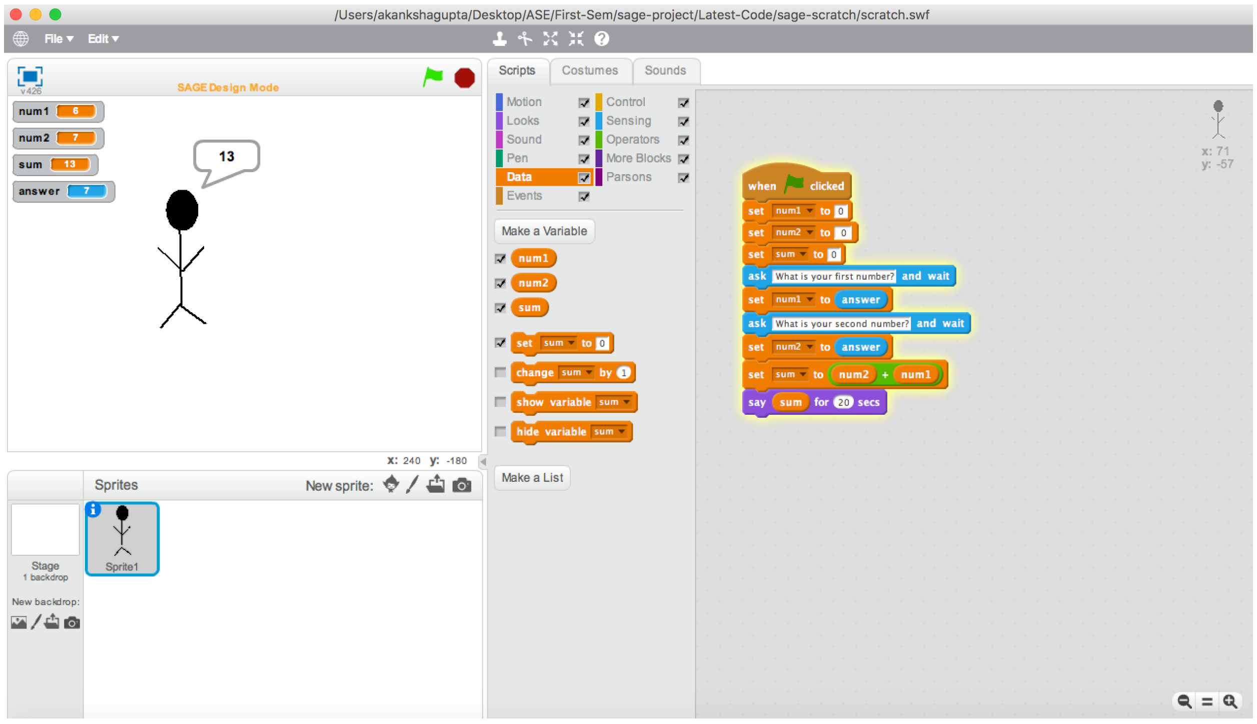
num1 6
num2 0
sum 0
answer 6

What is your second number?

7

```
when [green flag] clicked
set num1 to 0
set num2 to 0
set sum to 0
ask What is your first number? and wait
set num1 to answer
ask What is your second number? and wait
set num2 to answer
set sum to num2 + num1
say sum for 20 secs
```

It then shows the final sum on the stage.



```
when [flag] clicked
set num1 to 0
set num2 to 0
set sum to 0
ask What is your first number? and wait
set num1 to answer
ask What is your second number? and wait
set num2 to answer
set sum to (num2 + num1)
say sum for 20 secs
```

7.0 Code Repository

GitHub (Github, 2016) was used as the code repository for the coding for Parson's on Scratch Editor. We created a branch forked from from Sage(Jeff Bender,2015 and Jairo Pava, 2016).

Link to GitHub:
https://github.com/cu-sage/sage-scratch/tree/parsons

Wiki Link 1
https://github.com/cu-sage/sage-scratch/wiki/Parson's-Programming

Wiki Link 2
https://github.com/cu-sage/sage-scratch/wiki/Score-Metrics:-Parsons-Programming

8.0 Next Steps

1. Integrate points system with the existing functionality

2. Send stage and script data to assessment server for assessment and displaying results

3. Provide feature during designing of puzzle to add problem statement and hints.

4. Score metrics to evaluate Parson's puzzle results

5. Feedback option for student to share comments/ feedback back to the teacher

9.0 References

Bender, J. (2015). Developing a Collaborative Game-Based Learning System to Infuse Computational Thinking within Grade 6-8 Curricula.

Parsons, D. and Haden, P. (2006). Parson's Programming Puzzles: A Fun and Effective Learning Tool for First Programming Courses. In Proc. Eighth Australasian Computing Education Conference (ACE2006), Hobart, Australia. CRPIT, 52. Tolhurst, D. and Mann, S., Eds., ACS. 157-163.

J. Helminen, P. Ihantola, V. Karavirta, and L. Malmi. How do students solve parsons programming problems?: An analysis of interaction traces. In Proceedings of the Ninth Annual International Conference on International Computing Education Research, ICER '12, pages 119–126, New York, NY, USA, 2012. ACM.

L. Seiter and B. Foreman. Modeling the learning progressions of computational thinking of primary grade students. In Proc. Ninth annual Int'l. ACM Conf. on Computing Educ. Research - ICER '13, page 59, New York, New York, USA, Aug. 2013. ACM Press.

(2016, January 25). Retrieved from Scratch - Imagine, Program, Share: https://scratch.mit.edu/

Denny, P., Luxton-Reilly, A. and Simon, B. (2008). Evaluating a new exam question: Parsons problems. Fourth International Workshop on Computing Education Research(ICER '08), Sydney, Australia, 113-124

Barr, V., & Stephenson, C. (2011, March). Bringing Computational Thinking to K-12: What is Involved and What is the Role of the Computer Science Education Community? ACM InRoads, pp. 48-54.

Brennan, K., Balch, C., & Chung, M. (2014). Creative Computing. Harvard Graduate School of Education.

McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagen, D., Kolikant, Y., Laxer, C., Thomas, L., Utting, I., & Wilusz, T. A Multi-Institutional, Multi-National Study of Assessment of Programming Skills of First-year CS Students. SIGCSE Bulletin, 33(4).125-140, 2001.

Moreno-Leon, J., Robles, G., & Roman-Gonzalez, M. (2015, September). Dr. Scratch: Automatic Analysis of Scratch Projects to Assess and Foster Computational Thinking. Revista de Educación a Distancia.
Danielle S. Mc.Namara , Ville Karavirta (2011). Two-Dimensional Parson's Puzzles: The Concept, Tools, and First Observations. Aalto University, Finland

Danielle S. Mc.Namara , G. Tanner Jackson, Art Graesser. Intelligent Tutoring and Games (ITag) . Aalto University, Finland

Adobe. (2016, May). Adobe. Retrieved May 2016, from Adobe Flash Player: https://get.adobe.com/flashplayer/

What is the Role of the Computer Science Education Community? ACM InRoads.
Bender, J. (2015). Developing a Collaborative Game-Based Learning System to Infuse

Github. (2016, May). Retrieved from Github: https://www.github.com

MIT Lifelong Kindergarten Group. (2016, May). Retrieved from Scratch - Image, Program, Share: https://scratch.mit.edu/

MIT Lifelong Kindergarten Group. (2016, May). Github. Retrieved from Scratch 2.0 editor and player: https://github.com/LLK/scratch-flash