CodeVA Python SOL Development: Design Document

# Computer Science SOL Development

# Sponsor: CodeVA

## Department: Computer Science

#### 

#### Faculty Advisor: Dr. Budwell

#### CodeVA Project Leads: Bryan Wallace, Rebecca Dovi

### Design and Implementation Team:

#### Kenneth Richardson, Benjamin Napier, Seth Vickers, Dakota Brown, Hunter Frostick

## 

## Architecture/High-Level Design

Currently, SOLs for computer science have been created for K-12 education. There is very little supporting documentation for teachers to figure out how to teach computer science SOLs. Many teachers in K-12 education are not familiar with computer science and must be educated before they can teach computer science to their students. CodeVA offers teacher-training seminars during the summer for VA public school teachers. The facilitators that lead these seminars currently use CodeVA-created presentation slides and Scratch exercises.

In regards to the teacher-training aspect of our work with CodeVA, our team will need to:

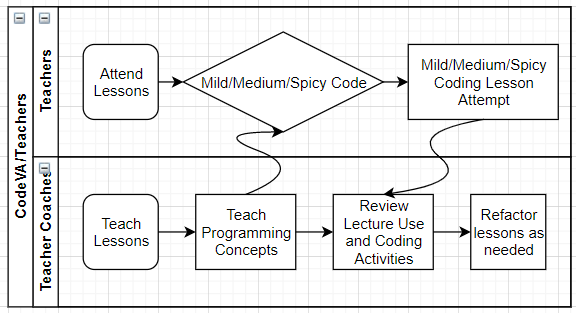
* Review the current teacher-training material (All members)
* Develop new or improve upon existing Scratch exercises to increase their rigor and effectiveness (3 members)
* Develop Python exercises to match the Scratch exercises for teachers who are more advanced or want to help their student with text-based languages (2 members)
* Create presentation slides, derived from the existing Scratch-related slides, to accompany the Python exercises (2 members)
* Improve upon the current presentation slides to make them more effective, clear, and streamlined (All members)

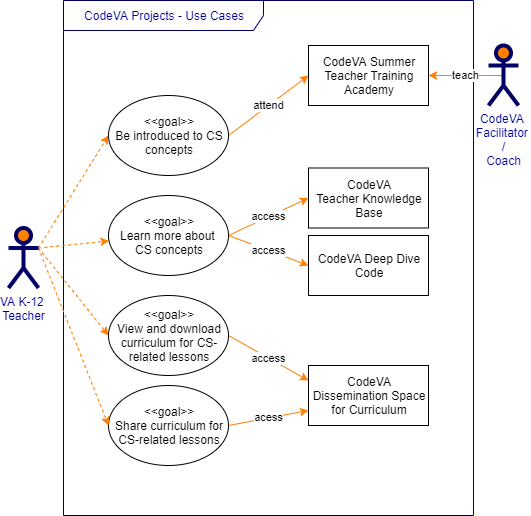
Additionally, CodeVA has other projects that we will be working on in the Spring. These projects are:

* AP Computer Science Principles: create an online AP CSP course and materials for Virtual VA
  + Help to create practice programs for learning and the curriculum
  + Help in whatever additional needs are asked for by the project lead
* Deep Dive Code: create a stand-alone course to introduce teachers to coding fundamentals more deeply
  + Create lessons that address the tools of programming
  + Create projects that correlate with lessons
  + Lessons in both scratch and python
* Dissemination Space for Curriculum: create a digital space to make curricular resources available for teachers to access and download
  + Have space to collect some sort of data ( i.e. email)
  + User submits and then be brought to page to download...etc ( not sent an email)
* Teacher Knowledge Base: help in authenticating and fleshing out CodeVA’s knowledge base of computer science concepts and ideas.
  + Curate resources for specific topics to explain and learn for teacher facing audience using the AP CS Principles course book
* CityScape: Creating a series of lessons using the client-side library P5.js.

## Detailed Design

* Swim Lane Diagram



* *Presentation Layer Designs*
  + Storyboards Presentation: <https://docs.google.com/presentation/d/1oeYGBxSMSNd4qKpi5R6VKQN6WoicnEvK3boiZ8lMYGQ/edit?usp=sharing>
* Use Case Diagram  
  

## Deep Dive Code

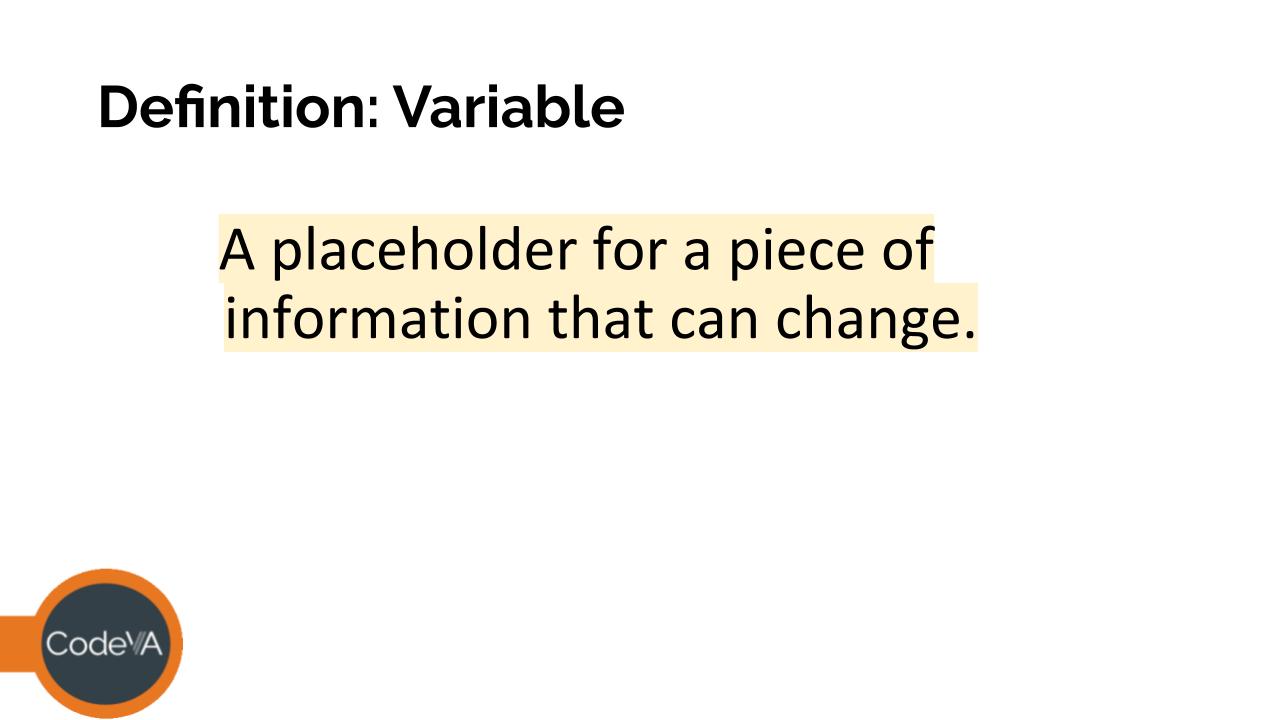
**Broad Overview:**

* Effortlessly introduces fundamental programming concepts using both block-based and text-based languages
* Each lesson is available in both block- and text-based language, making transition from one to another seamless
* Teachers can use lessons as a guide or as a stand-alone lesson plan

### Scope:

* Each lesson introduces and defines a key fundamental computer science concept, using either a block-based (Scratch) or text-based (Python) programming language to assist
* Lessons begin with definitions and a warm-up exercise (a program) that demonstrates the CS concept being taught
* Learners will then gain hands-on experience via a coding lab in which they are guided through the steps of writing code to demonstrate their newfound CS conceptual knowledge

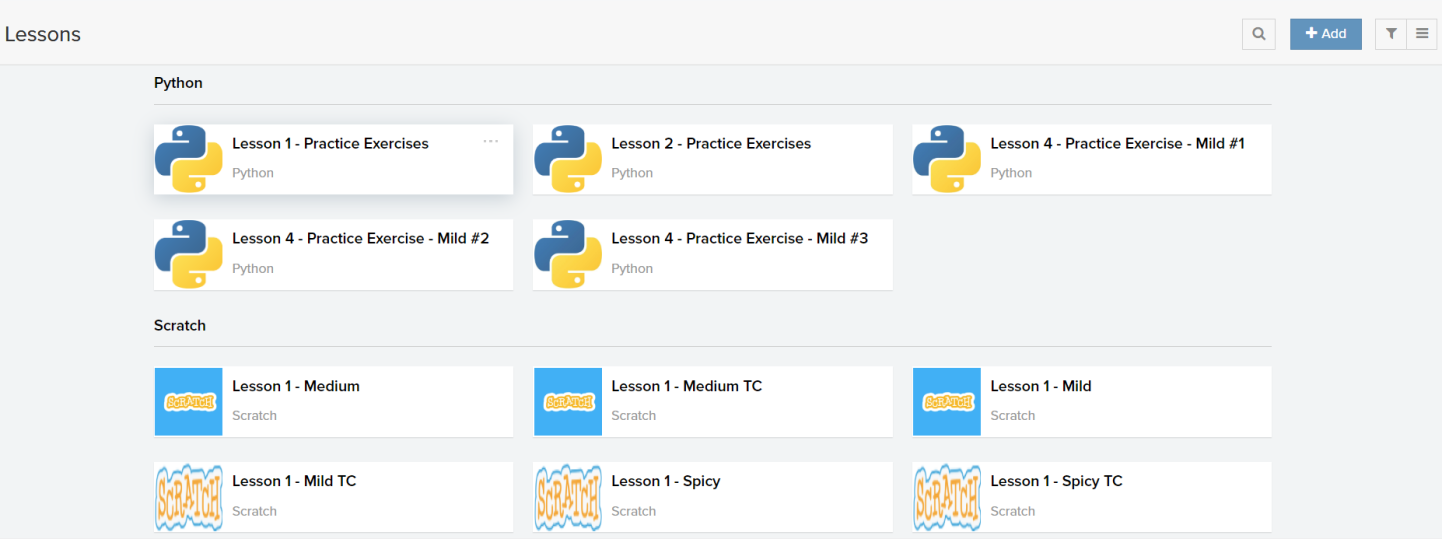
### First Look:

* A slide from an already existing CodeVA lesson on *Input, Output, and Variables*, defining what a variable is:

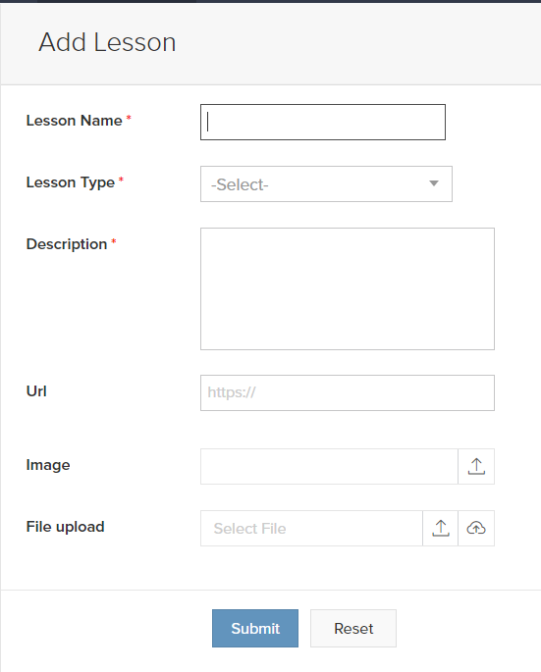
## User Interface: Dissemination Space

### Broad Overview:

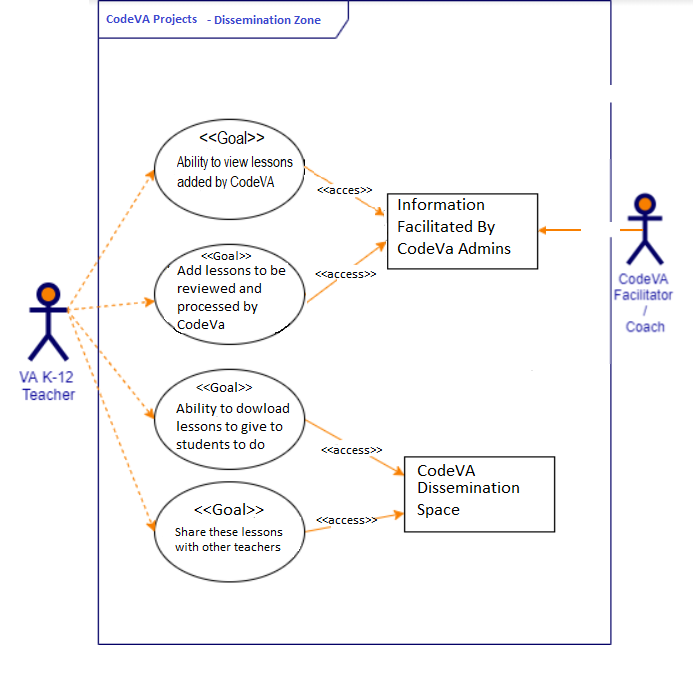
* CS educational resources and content openly available to all, not just students and educators
* Allows teachers to potentially submit their own material
* Collaborations between universities, public schools, and even communities possible
* Our Lessons Plan page is the user interface the teachers will see. This is where we have added our python and scratch lessons with descriptions and links to download. This is the space where teachers will also be able to add their own lessons that a CodeVa facilitator will review and add if approved.



* This Add Lesson space is where you will be guided when adding a lesson. Right now we only were tasked to create python and scratch lessons so it is split up into adding a lesson into either category. You will be asked to provide a link or a file upload to access the lesson and also a description of the lesson as well as a lesson plan on how to complete the lesson provided.



* + Dissemination Space Diagram



## AP Computer Science Principles Lessons

### Broad Overview:

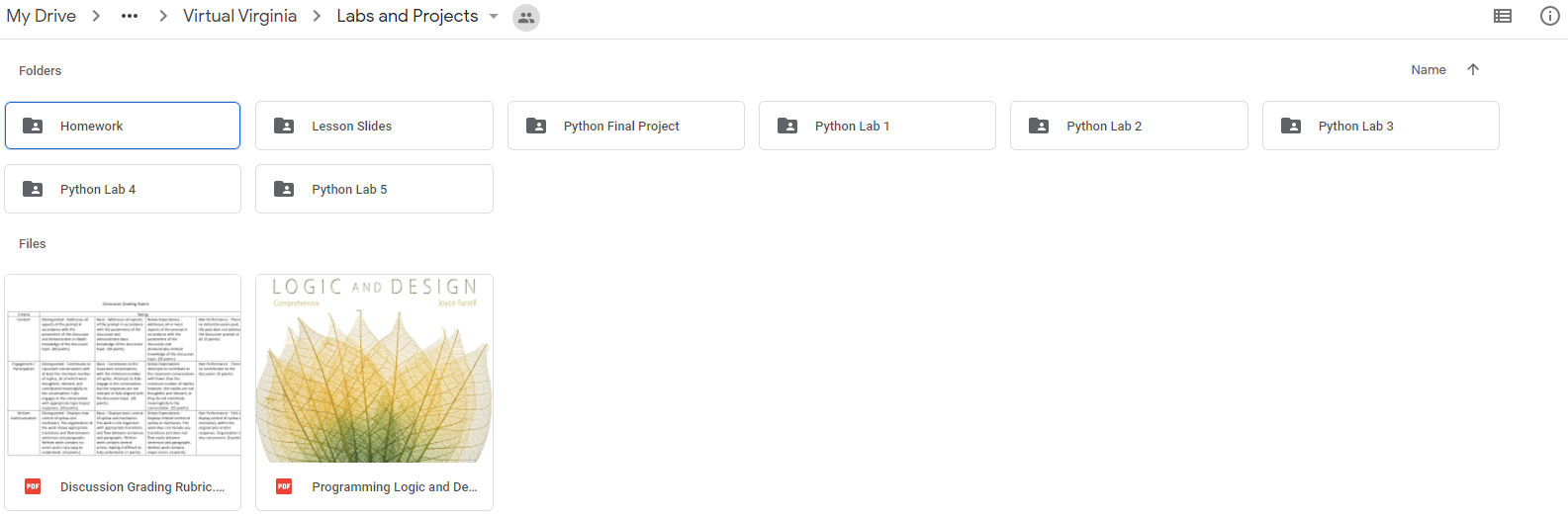
* Python labs, lessons, and homework assignments designed for use by Virtual Virginia's AP Computer Science Principles course
* Introduction to Python, programming logic, and object-oriented programming
* Fun projects, such as "Your Name in Code," to make learning Python an interactive process

### Scope:

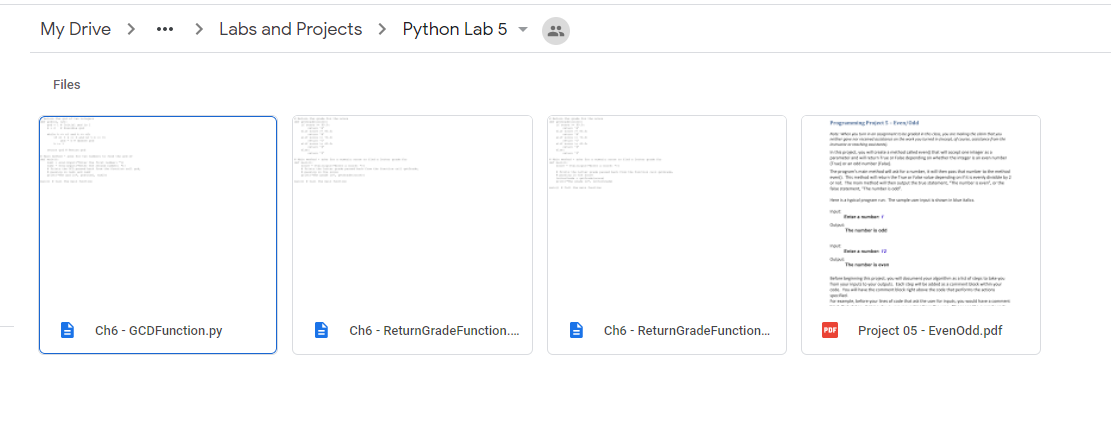
* Each Python lab and module project contains a presentation with directions that take students from never having tried programming to being able to create a final project which includes multiple functions using parameters, loops, and conditionals.
* Each lab contains an example completed project for teachers to use as a baseline along with rubrics to suggest how to grade the projects.

### First Look:

* A shared google drive organized so that the educators of the AP Computer Science Principles course can easily find the lessons they are looking for.



* Each lab lesson contains a project, a rubric to grade the project, and code skeletons for how to do the project.



## CityScape P5.js Lessons

### Broad Overview

* + P5.js provides a visual learning experience for text-based programming
  + By introducing simple, yet essential, programming skills in the form of lessons with visual representation, we compound information and experience in the creation of a CityScape
  + These lessons will ease the transition from a visual, block-based programming language like Scratch to a text-based one

### First Look

* + Without having to worry about setting up any sort of integrated development environment, students utilize text-based programming and learn javascript all inside of their favorite browser!

|  |  |  |  |
| --- | --- | --- | --- |
| **Lesson Topic** | **Narrative** | **Lesson Link** | **Time** |
| 0: Introducing p5js | Students are familiarized with the web application of the p5js web editor and the accompanying resources for this tool. | [Lesson 0](https://docs.google.com/document/d/1z57sY1pPChqckrQxWIp6XzEuPbUXT3RSq_E3GUEXKe4/edit?usp=sharing) | 60 min |
| 1: The Canvas and Drawing Lines | Covers preparing canvas size and the Cartesian coordinate system for this tool. Students will use lines to connect several (x,y) points after planning on grid paper. | [Lesson 1](https://docs.google.com/document/d/1TU3kivjBkLGrHlEAblTM_qY-mwCuzIYjQBxzkl8_k0w/edit?usp=sharing) | 60 min |
| 2: Drawing Shapes | Covers the use of primitive shapes and beginShape(). Students will use these tools to render shapes they have planned out on grid paper. | Lesson 2 | 60 min |
| 3: Color & Layering | Covers the use of fill() and the concept of layered rendering. | Lesson 3 | 60 min |
| 4: Variables | Covers the use of variables and their use in setup() as global variables. | Lesson 4 | 30 min |
| 5: Objects | Covers the concept of objects. These are unordered, unindexed, and keyed. These should be utilized for complex variables. | Lesson 5 | 30 min |
| 6: Loops and Variable Updates | Introduces the concept of loops which can be used to draw static objects and updating variables (infinitely). | Lesson 6 | 60 min |
| 7: Conditionals | Introduce the concept of conditionals. Very important to stop all of those pesky infinite updates. | Lesson 7 | 60 min |
| 8: Functions | Covers the use of functions for making generic code. | Lesson 8 | 60 min |
| 9: Wrapper Functions | Introduces the concept of calling a function with a function. | Lesson 9 | 30 min |

