

# User Documentation

## Sprint Release 11

### Student View

#### Loading the Canvas Assignment

Once the application is running on your device and the development setup has been done, you can navigate to the Canvas assignments page, select **CodingExam Assignment 1** which will load the **CodingExam** application and pull the exam questions already existing in the database (Note that there are some default questions in development mode). The following screenshots are based on sample data created by a user, but the application should look generally like the screenshots below when data is added.

Question 1
Example Question 1
<div>My submission</div>

Question 2
Example Question 2
<p><input type="radio"/> a</p> <p><input checked="" type="radio"/> b</p>

Submit

#### Multiple Choice Component

The user will be presented with a question and four different options for them to select. Once selected, the component should look like the screenshot below.

Question 2
Example Question 2
<p><input type="radio"/> C#</p> <p><input checked="" type="radio"/> Java</p> <p><input type="radio"/> TypeScript</p> <p><input type="radio"/> Fortran</p>

## True False Choice Component

The user will be presented with a question and then true or false options for them to select. Once selected, the component should look like the screenshot below.

Question 3
Example Question 3
<input checked="" type="radio"/> True
<input type="radio"/> False

## Short Answer Component

The user will be presented with a question prompt and a blank text entry box to answer the question with. Once text is entered, the component should look like the screenshot below.

Question 1
Example Question 1
<div>My submission</div>

## Code Editor Component

The second to last question visible should be a code editor question. This will allow for the user to write only Java code and have syntax highlighting for it appear. The user is not able to compile the code with the component. Once code has been entered the syntax highlighting should look like the screenshot below.

Question 4
Example Question 4
<pre>1 print("Hello World!")</pre>

## Parson's Problems Component

The last question visible should be the parson's problem, the user should see a blue **Submit** button that will save and submit their answers to the database. The parson's problems code snippets should look like the screenshot below.

Print "Hello World" 3 times

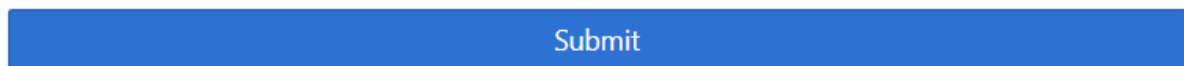
Drag from here

```
for(int i; i< 3; i++){
    print("Hello");
    print("World");
}
```

Construct your solution here

## Submit Button Component

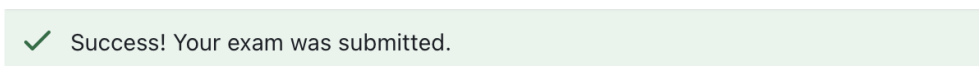
At the end of the CodingExam Application, the user should see a blue **Submit** button that will save and submit their answers to the database. The button should look like the screenshot below.



The user can select an answer and hit the **Submit** button to submit their answer to the server. The application will then display a message at the top of the screen showing the success of the submission to the database. It should appear like the screenshot below.

## Valid submission

They should also see a green box message appear at the bottom of the exam



## Reloading the CodingExam Application

Once the user has submitted their response by clicking the **Submit** button, the data should be saved and stored in the database. Upon reloading the webpage, the page should repopulate the questions with their responses saved in the database. Each question should be disabled, and feedback boxes should appear.

Question 1
<p>Example Question 1</p> <div>My submission</div> <p>Feedback:</p> <div></div>

# Instructor View

## Loading the Application

When the user is an instructor and opens the application on a particular assignment, they will be greeted with a screen showing a list of the students who have submitted the assignment. The assignment from the Student section will be used for this section. The screen should look like the screenshot below.

[Back](#) [Create Exam](#) [Grade Exams](#)

### Student List

Example Learner

## Clicking a Student from the List

When a student's name is clicked on by the user, the application will load the student's exam submission and display it for the instructor. The following screenshots will be based on the **Example Learner** student. For the exam from the student section of this document, the screen loaded by the application should look like the screenshot below.

**Question 2**

Example Question 2

☐ C#

☒ Java

☐ TypeScript

☐ Fortran

Feedback:

## Clicking the Back Button

The back button at the top of the page can be clicked to return to the list of students for the current exam. Note that any feedback entered into the boxes on the page will not be saved when the back button is clicked. The button should look like the screenshot below.

[Back](#)

## Leaving Feedback

Each question loaded in the instructor view should also have a text box loaded below it. This text box is for leaving feedback. The instructor can write feedback for each of the questions the student answered in the appropriate box below the question. If feedback has already been left, the boxes should contain the feedback that was left, and if feedback has not been left yet, the boxes should appear blank. A question and feedback box should look like the screenshots below if filled in.

**Question 2**

Example Question 2

☐ C#

☒ Java

☐ TypeScript

☐ Fortran

Feedback:

Feedback for question 2

## Submit Feedback Button

Once the instructor has left the feedback they wish to leave, the submit feedback button can be clicked to submit the feedback to the database. After clicking the button, the application's screen should return to the initial student list view. The button should look like the screenshot below

Submit Feedback

## Viewing Given Feedback

After the instructor has left feedback for a particular student's exam, the same student can be clicked on again to view the feedback that was left. The loaded feedback should look like the screenshot below.

**Question 2**

Example Question 2

☐ C#

☒ Java

☐ TypeScript

☐ Fortran

Feedback:

Feedback for question 2

## Creating an Exam

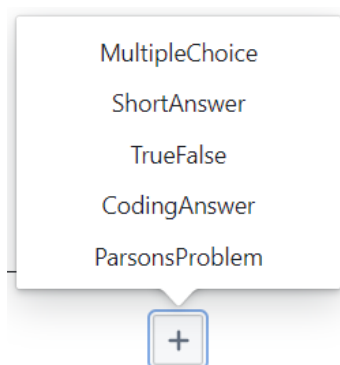
The instructor can now create an exam by clicking the “Create Exam” button in the base view. This will present them with the following view, where they can create each question and see a preview of the exam.

Back

+

Done

From here, clicking the “+” will present the user with a list of question options to create, as follows.



After all questions have been created or edited, the “Done” button must be clicked to submit the changes to the exam to the database. If the “Done” button is not pressed, the edits to the exam will not be saved.

Done

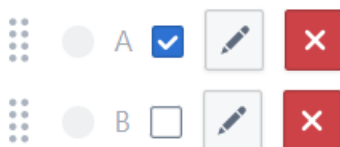
## Creating a Multiple-Choice Question

After clicking the “MultipleChoice” option, the user will be prompted with the following. Here, they can enter the question text and add answer choices using the plus button. Additionally, the box next to points can be filled in to set the points possible for the question. This applies to all question types.

When adding an answer, the green check can be pressed to add the answer, and the red “x” can be pressed to remove the answer.



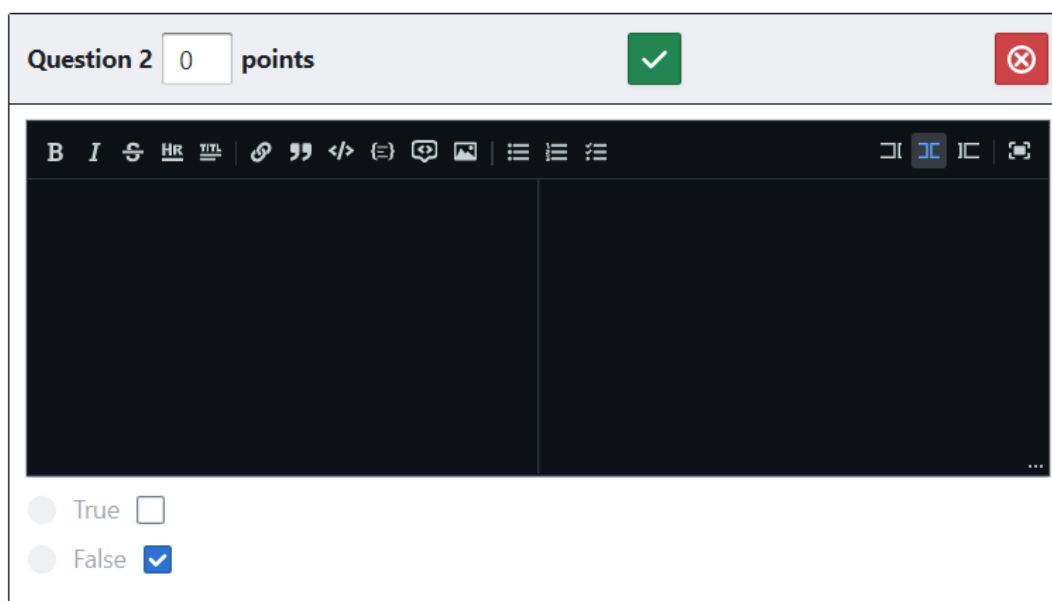
Once multiple answers have been added to the question, they can be reordered by dragging the handle on the left side of each answer.



Additionally, the pencil button can be pressed to edit the answer’s text, the red “x” button can be pressed to remove the answer, and the checkbox can be checked to mark the correct answer.

## Creating a True False Question

After clicking the “TrueFalse” option, the user will be prompted with the following. The question text can be set in the text box, the correct answer can be selected by clicking the checkboxes next to the two options, and clicking the green checkmark will finish creating the question.



## Creating a Short Answer Question

After clicking the “ShortAnswer” option, the user will be prompted with the following. Here, they can enter the question text in the text box, and the green check button can be pressed to finish creating the question.

The screenshot shows a question editor interface for 'Question 3'. At the top, there is a header bar with 'Question 3', a text input field containing '0', and the word 'points'. To the right of this bar are a green checkmark button and a red 'X' button. Below the header is a large text editor area with a dark background and a toolbar containing various icons for text formatting (bold, italic, underline, strikethrough, link, unlink, quote, code, list, indent, outdent) and media insertion (image, video, audio). The editor area is currently empty.

## Creating a Coding Answer Question

After clicking the “CodingAnswer” option, the user will be prompted with the following. Here, they can enter the question text. They can also select the programming language they want the answer to be written in with the “Select language”. The green check button can be pressed to finish creating the question.

The screenshot shows a question editor interface for 'Question 4'. It has a similar layout to the previous one, with a header bar containing 'Question 4', a text input field with '0', and the word 'points'. To the right are a green checkmark button and a red 'X' button. Below the header is a text editor area with a dark background and a toolbar. Underneath the editor is a dropdown menu labeled 'Select language...'. Below the dropdown is a horizontal bar with the number '1' on the left.

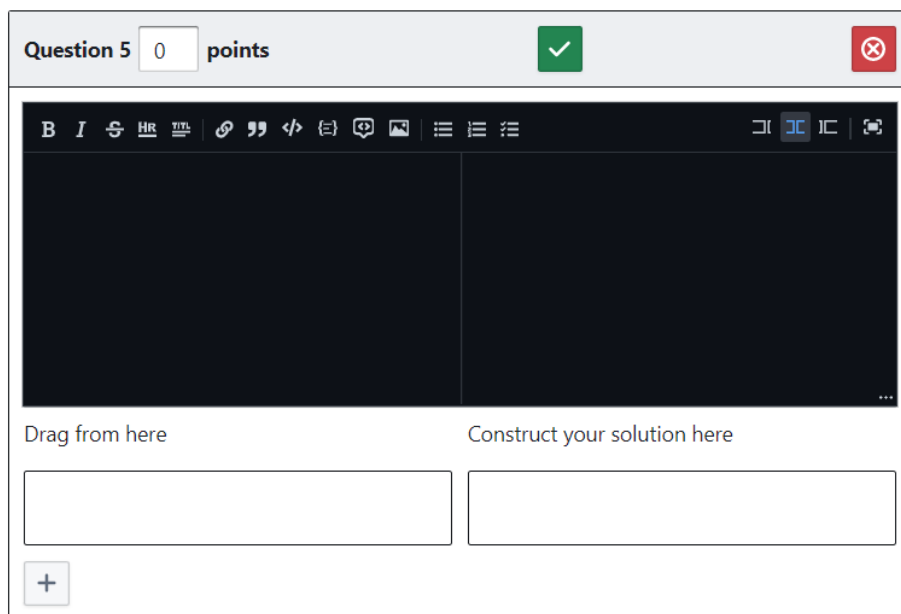
The following languages are currently available to select from the language box.

This screenshot shows a close-up of the 'Select language...' dropdown menu. The menu is open, displaying a list of programming languages: 'java', 'python', and 'csharp'.



## Creating a Parsons Problem Question

After clicking the “Parsons Problem” option, the user will be prompted with the following. Here, they can enter the question text in the box, as well as add options for the left column by clicking the plus button.



The screenshot shows a question creation window titled "Question 5" with a score of "0 points". It features a rich text editor with a toolbar containing icons for bold, italic, link, unlink, list, and other formatting options. Below the editor, there are two columns: "Drag from here" and "Construct your solution here", each with a text input box. A plus button is located at the bottom left of the interface.

Clicking the plus button will bring up the following box, which can be used to add new options to the left column. Note that these options will not appear until the exam has been fully created and the page has been reloaded.



The screenshot shows a single-line text input box for adding options. To the right of the input box are two buttons: a red button with a white "x" icon and a green button with a white checkmark icon.

## Editing a Question

At any time, the user can click the orange note icon to edit a question. This will bring back up the question creation screen for that type of question. Clicking the green checkmark again will save the edits to the question.



## Deleting a Question

At any time, the user can click the red “x” to the upper-right of a question, which will delete the question from the exam. Note that this will only delete the question if it has not been added to the exam with the “Done” button already. It cannot delete existing questions from the exam



## Grading Exams

The instructor can also grade exams by clicking on the “Grade Exam” button on the InstructorView. This will show you the following screen.

Back
Grade

Question 1  
 Question 2  
 Question 3  
 Question 4  
 Question 5  
 Question 6  
 Question 7  
 Question 8

**Example Question 1** `def myfunction: ```

Student	Submission	Feedback	Grade
Example Learner	My submission	Good work	0 / 5

Users can click entries in the list of questions on the left-hand side to go change which question is being viewed. Users can also edit the value in the “Feedback” box to change the current feedback for a student’s response. Finally, users can edit the “Grade” box to change the student’s grade for that question submission. Clicking the “Grade” button will submit the grades that are currently in the database for the assignment to Canvas. It will also display a message showing whether it was successful or not. In the picture below, the submission was unsuccessful. Note that this will be the only response if the application is not running in Canvas.

## Grade Submission Unsuccessful