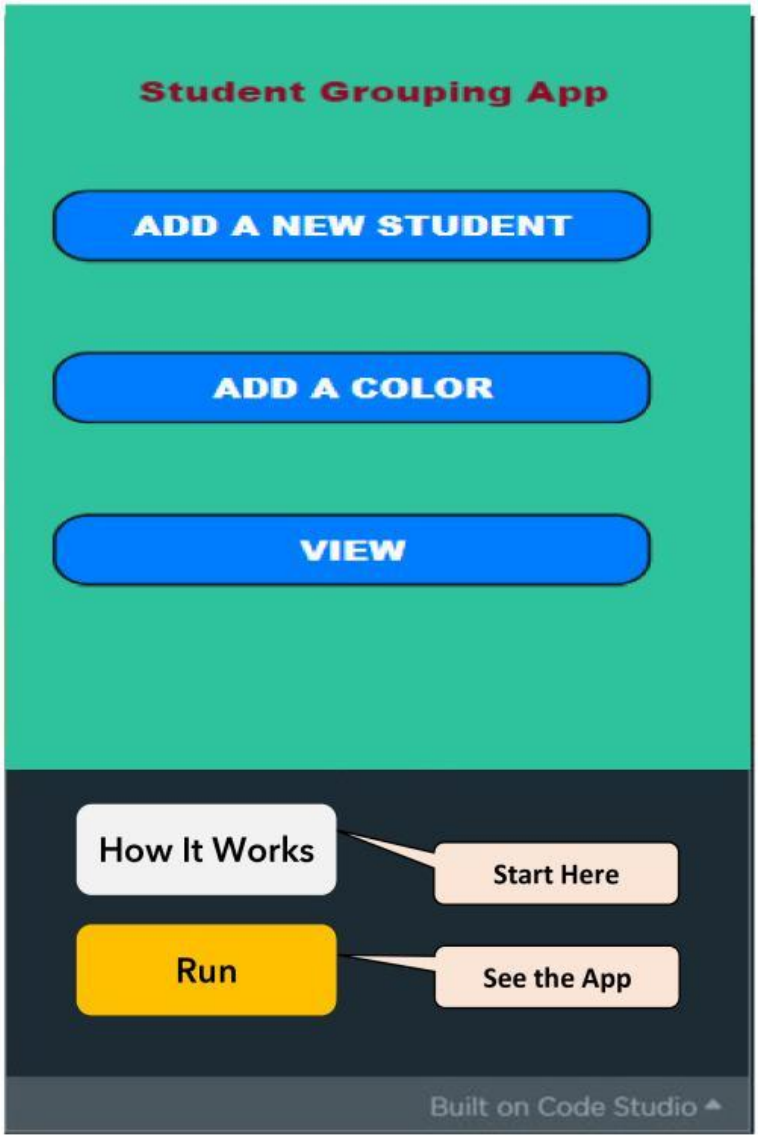
Project 83

DP EDUCATION

Coding School



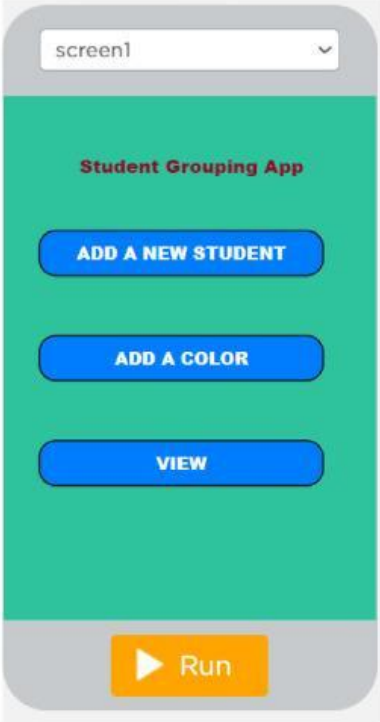
LIVEWORKSHEETS

❖ Let's create an app to divide 20 students into 4 groups using Array.

This app is used to divide up to 20 students into 4 groups. 14 students are provided by an array and the remaining students can be added to the array.

Students should be divided into 4 groups according to their ID.

Each group has a separate screen and there are 5 labels in that screen. Another array is used to change the color of the labels.



❖ All the design related to the app has been given to you.

❖ Give two separate arrays for student ID and name. Also create an array for color. The color array consists of color codes. Create two variables as len and lenColors and give them the length of the studentId array and colors array.

```
var studentsId = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14];
var studentsName = ["Amila", "Amal", "Amara", "Binura", "Bawantha", "Chamara", "Chamika", "Dasun", "Darshana", "Danushka", "Eshan", "Isuru", "Sumudu", "Oilki"];
var colors = ['#801616', '#08891b', '#c0cb10', '#1810b6', '#801616', '#08891b', '#c0cb10', '#1810b6', '#801616', '#08891b', '#c0cb10', '#1810b6', '#801616', '#08891b'];
var len = studentsId.length;
var lenColors = colors.length;
```

❖ Let's see how to create the connection between Screen.

LIVEWORKSHEETS

```
onEvent("buttonNextA", "click", function() {
  setScreen("screenB");
});
onEvent("buttonNextB", "click", function() {
  setScreen("screenC");
});
onEvent("buttonNextC", "click", function() {
  setScreen("screenD");
});
onEvent("buttonNextD", "click", function() {
  setScreen("screenA");
});
onEvent("btnHome", "click", function() {
  setScreen("screen1");
});
```

- When "ADD A NEW STUDENT" button is clicked, you should go to screen 2 and if the error text labels are shown there, you should hide them.

```
onEvent("btnAddNewStudent", "click", function() {
  hideElement("errorMsg1");
  hideElement("errorMsg2");
  setScreen("screen2");
});
```

- When "ADD A COLOR" button is clicked, you should go to screen3.

```
onEvent("btnAddColor", "click", function() {
  setScreen("screen3");
});
```

LIVEWORKSHEETS

Screen2

- Screen2 is used to add a new student. A new value is added to the "studentsId" array and "studentsName" array.

- When back button is clicked, you should go back to screen1.

```
onEvent("buttonBack1", "click", function() {
  setScreen("screen1");
});
```

(screen2)

- When the Add button is clicked, the ID should be added to the "studentsId" array and the name should be added to the "studentsName" array.

- Create 2 variables as “newId” and “newName”. Then if the length of the "studentsId" array is equal to 20, the "errorMsg2" label should be shown. Because this app is designed for a maximum of 20 students.

```
onEvent("buttonAddNewStudent", "click", function() {
  var newId;
  var newName;
  if(len == 20){
    showElement("errorMsg1");
  }
```

- In the Else part, create a variable as "idCheck" and give its value as 1.

- Assign the new student's id and name to two variables "newId" and "newName".

- Create a for loop and check whether the id of the new student has been stored in the "studentsId" array.

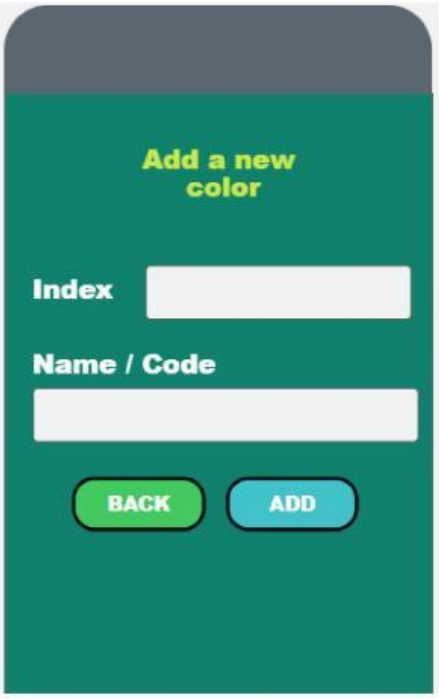
LIVEWORKSHEETS

- If so, set the value of the “idCheck” variable as 0.
- Now if the value of “idCheck” is 0 in an if condition, show the “errorMsg2” label.
- This will check if the id of the new student is in the array we provided earlier. If so, an error massage will be displayed in red by the "errorMsg2" label.
- If the student to be added is not in the previous array, the new student can be added to the array. Do related coding in the Else part.

```
else{
  var idCheck = 1;
  newId = getText("inputNewStudenID");
  newName = getText("inputNewStuName");
  for(var i = 0; i < len; i++){
    if(studentsId[i] == newId){
      idCheck = 0;
    }
  }
  if(idCheck == 0){
    showElement("errorMsg2");
  }
  else{
    hideElement("errorMsg2");
    appendItem(studentsId, newId);
    appendItem(studentsName, newName);
    len = studentsId.length;
    setScreen("screen1");
  }
}
});
```

LIVEWORKSHEETS

Screen3



- Screen3 adds a new color to the "colors" array.

- When back button is clicked, you should go back to screen1.

```
onEvent(▼"buttonBack2", ▼"click", function() {
  setScreen(▼"screen1");
});
```

- When the Add button is clicked, the new color should be added to the position given to the Index in the "colors" array.

- Here the new color is added to the array at the location we provide. For that, the code block in the variables of the Tool box is used.

(screen3)

```
insertItem(list, 2, "c");
```

Value is the index to be added

```
insertItem(list, 2, "c");
```

The name of the array.

The newly added value

- Create 2 variables as “newIndex” and “newColor”. Then, if the length of the "colors" array is equal to 20, the "errorMsg3" label should be shown. Because this app is designed for a maximum of 20 students.

- In the Else part, assign the index and the name/code of the color to two variables "newIndex" and "newColor"

- According to the given index, assign the new color to the "colors" array.

LIVEWORKSHEETS

```
onEvent("buttonAddNewColor", "click", function() {
  var newIndex;
  var newColor;
  if(len == 20){
    showElement("errorMsg3");
  }
  else{
    newIndex = getText("inputIndex");
    newColor = getText("inputColorName");
    insertItem(colors, newIndex, newColor);
    setScreen("screen1");
  }
});
```

- When the view button is clicked, students should be grouped into 4 anuwa groups according to the student's id. And a color should be set for the labels.

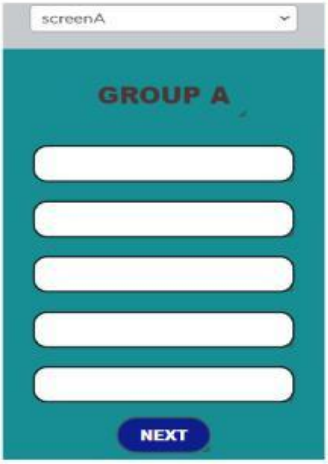
Let's consider how to group according to Student Id.

Here children are divided into 4 groups according to student id. The Id is divided by 4 and divided into 4 groups according to the reminder.

If Reminder = 1 then group A,
If Reminder = 2 then group B,
If Reminder = 3, then group C
As group D if Reminder = 0

Children should be divided into 4 groups.

- Children of group A should be displayed on screenA, children of group B should be displayed on screenB, children of group C should be displayed on screenC and children of group D should be displayed on screenD.



(screenA)

- screenA, screenB, screenC and screenD have 5 labels each and their id is as follows.

screenA: labelA1, labelA2, labelA3, labelA4 and labelA5
screenB: labelB1, labelB2, labelB3, labelB4 and labelB5
screenC: labelC1, labelC2, labelC3, labelC4 and labelC5
screenD: labelD1, labelD2, labelD3, labelD4 and labelD5

LIVEWORKSHEETS

- Let's create the "viewGrouping" function !
- Create 4 variables as a, b, c, d to get id of labels. Give 1 as its value.
- Create a for loop in the function and increase the i variable of the for loop from i = 0 to the size of the length of the "studentsId" array one by one to check the element of the "studentsId" array one by one. In the for loop, divide the value of "studentsId[i]" by 4 and get its reminder.
- Divide into groups according to the value of the reminder.
- Use the setProperty block to set values to the text of the labels.

```
function viewGrouping() {
  var a = 1;
  var b = 1;
  var c = 1;
  var d = 1;
  for( var i =0; i < len; i++){
    if( studentsId[i] % 4 == 1){
      setProperty( "labelA"+ a , "text", "ID : " + studentsId[i] + " Name : " + studentsName[i] );
      a++;
    }
    else if( studentsId[i] % 4 == 2){
      setProperty( "labelB"+ b , "text", "ID : " + studentsId[i] + " Name : " + studentsName[i] );
      b++;
    }
    else if( studentsId[i] % 4 == 3){
      setProperty( "labelC"+ c , "text", "ID : " + studentsId[i] + " Name : " + studentsName[i] );
      c++;
    }
    else{
      setProperty( "labelD"+ d , "text", "ID : " + studentsId[i] + " Name : " + studentsName[i] );
      d++;
    }
  }
}
```

LIVEWORKSHEETS

Let's create the "colorRow" function

- Let's create the "colorRow" function

- Create 4 variables as a, b, c, d to get id of labels. Give 1 as its value.

- Create a for loop in the function and to check the element of the "colors" array one by one, increase the i variable of the for loop from i = 0 to the size of the length of the "colors" array one by one. In the for loop, divide the value of "colors[i]" by 4 and get its reminder.

- Use the setProperty block to set values to the background-color of the labels.

```
function colorRow() {
  var a = 1;
  var b = 1;
  var c = 1;
  var d = 1;
  for(var i = 0; i < lenColors; i++){
    if( i % 4 == 1 ){
      setProperty("labelA" + a, "background-color", colors[i]);
      a++;
    }
    else if( i % 4 == 2 ){
      setProperty("labelB" + b, "background-color", colors[i]);
      b++;
    }
    else if( i % 4 == 3 ){
      setProperty("labelC" + c, "background-color", colors[i]);
      c++;
    }
    else {
      setProperty("labelD" + d, "background-color", colors[i]);
      d++;
    }
  }
}
```

- When the View button is clicked, call the "viewGrouping" function and the "colorRow" function.

```
onEvent("btnView", "click", function() {
  viewGrouping();
  colorRow();
  setScreen("screenA");
});
```

LIVEWORKSHEETS