Programming with B4X

Curriculum Plan for middle and high schools

Version 1.0, February 2021

Anywhere Software

Table of Contents

Anywhere Software

Introduction

Computer Science is more than programming, but programming is an absolutely central process for Computer Science. In an educational context, programming encourages creativity, logical thought, precision and problem-solving, and helps foster the personal, learning and thinking skills required in the modern school curriculum. Programming gives concrete, tangible form to the idea of "abstraction", and repeatedly shows how useful it is. (Computing At School Working Group, 2012)

In the following sections we will deal with the teaching of programming with the B4X language. Initially the questions that will concern us are:

- What's a problem?
- How can we clearly describe the solution to a problem?
- In what language does the computer "understand" the commands we give it?
- What is Algorithm
- Implementing a Computer Algorithm
- Programming in B4J

What pupils should know

The expected educational outcomes depend on the students' previous level of knowledge. Each teacher is recommended to adjust his expectations according to the age, background and cognitive level of his students. More general issues that the teacher should take into account are:

Algorithms

- An algorithm is a sequence of precise steps to solve a given problem.
- A single problem may be solved by several different algorithms.
- The choice of an algorithm should be influenced by the data structure and data values that need to be manipulated.
- The choice of an algorithm to solve a problem is driven by what is required of the solution [such as code complexity, speed, amount of memory used, amount of data, the data source and the outputs required].
- Familiarity with several key algorithms.
- The need for accuracy of both algorithm and data [difficulty of data verification; garbage in, garbage out]
- Different algorithms may have different performance characteristics for the same task.

Programs

Pupils should know how to write a program in B4J (B4X)

- A computer program is a sequence of instructions written to perform a specified task with a computer.
- Programs are developed according to a plan and then tested. Programs are corrected if they fail these tests.
- A well-written program tells a reader the story of how it works, both in the code and in human-readable comments
- Programming is a problem-solving activity, and there are typically many different
- programs that can solve the same problem.
- Variables and assignment.
- Programs can work with different types of data [integers, characters, strings].
- The use of relational operators and logic to control which program statements are
- executed, and in what order
- Simple use of AND, OR and NOT
- How relational operators are affected by negation
- Abstraction by using functions and procedures (definition and call), including:
- Functions and procedures with parameters.
- Programs with more than one call of a single procedure.
- Documenting programs to explain how they work.
- Understanding the difference between errors in program syntax and errors in meaning. Finding and correcting both kinds of errors.
- Manipulation of logical expressions, e.g. truth tables and Boolean valued variables.
- Lists
- Maps
- Files

Data

A pupil should understand how computers represent data:

- Information can be stored and communicated in a variety of forms e.g. numbers, text, sound, image, video.
- Introduction to binary representation [representing names, objects or ideas as sequences of 0s and 1s].
- The difference between constants and variables in programs.
- Difference between data and information.
- String manipulation

Table of teaching Items

In the table below will cover the most necessary aspects of programming with B4X. For every unit will provide:

- brief description of the corresponding theory,
- lecture slides in power point,
- examples of solved exercises for understanding,
- exercises to learn in different levels of difficulty

The total teaching time was calculated at **54** hours, but this also depends on the level of the students and can be redefined.

Lesson		**Details**	**Hours**
1	The B4X language	• Why B4X • Downloading and Installing B4J and Java • Customize environment	1
2	The meaning of the problem	• What is a problem • Ways to represent a problem	1
3	My first Program	• Create a new program • How to run a program • How to Save • The turtle	2
4	Variables and Range	• Int • Float • How to name a variable • Mathematical Operators • Assign Values to Variables • The log function • Strings	3
5	Designer	• Talking about Designer • Design the first Screen • Views: Labels, TextFields, Buttons, Panes • Saving forms	2
6	From Designer to Code	• Passing Values to Code • What happens when we write a value into a textField • "What happens when we push a button" – Events	1
7	Conditional Statement	• Boolean Variables • Logical Operators • If Statement • If-Else Statement • If-Else-Else If Statement • Nested If Statements	3
8	B4XPages	• Class – Object a brief discussion • What is a B4XPage • How to Create and Delete a B4XPage • Passing Values within Pages	2

Lesson		Details	Hours
9	Application 1	• "A simple Calculator" • With the help of teacher pupils creates a Calculator application with the 4 simple operations.	3
10	B4XViews	• What is a B4XView • Hot to create B4XViews • B4XViews events	1
11	Loops	• What are Loops? • Types of Loops in B4X • For – Next • For – Each • Do While	5
12	String manipulation	• String concatenation • String builder • String functions	2
13	Subroutines	• What is a subroutine • Declaring a Sub • Passing Values • Returning Values from a sub	4
14	XUI Views	• Creating Dialogs	2
15	Arrays	• One dimensional Arrays • Basic Operations with arrays • Linear search • Binary search • MAX – MIN item • Sorting with Bubble Sort • Sorting with Selection Sort	4
16	Lists	• What is a list? • Basic Operations with lists	2
17	Maps	• What is a map? • Basic Operations with maps	2
18	Files	• File location in B4J • File Methods	2
19	Application 2	In this app pupils will test their knowledge creating an application based in previous lessons.	10
20	From B4J to B4A	How to move an application to B4A	2
		Total Hours	**54**

Lesson 1 – Why B4X?

⏱ **1h**

Many new developers wonder in which programming language to invest their time and effort. Each programming language has advantages but also disadvantages that should be considered. Often the selection of a language also determines the subsequent evolution of the developer. Even more, when it comes to using language for educational purposes, there are individual elements that need to be considered.

So, the programming language must be:

- Modern and structured
- Be easy to learn for kids and new programmers.
- Provide an integrated development environment without confusing.
- To be able for the student to develop applications on different platforms like Windows, Android, IOS, Linux, Raspberry Pi, Arduino etc.
- Provide all modern data structures as lists, maps etc.
- To keep students interested by providing the possibility of developing different types of applications for example games etc.

The **B4X** programming language provides all the above features and is also a language for developing high-level applications which can be a springboard for future professional development. In addition, it has a very enthusiastic audience that gladly help in any kind of question.

For more information and material you can visit the website at https://www. b4x. com/learn. html

Installing B4X

The most updated information about installing will always be here: https://www.b4x.com/b4j.html

Lesson 2 – The meaning of the problem

⏱ **1h**

What students should know

- What is a problem?
- What is data?
- What is information?
- What steps need to be taken to resolve a problem.
- What is algorithm.

The concept of the problem

Every day in our lives we have problems. Simple problems of everyday life and often even bigger. As a problem we usually define a situation that we are experiencing and which makes it difficult for us to deal with it or solve (Cambridge, 2021).

When we are thinking about a problem these are steps become unconscious in our minds:

- Understanding the problem
- Searching for a solution or set of solutions.
- Choosing the right solution
- Implementing the solution
- Checking whether this solution had the desired results.

Understanding the problem

Understanding the problem is the first step in designing and solving a problem. It is a particularly critical stage because this will also affect the development of the solution. Includes the following steps:

1. Description of the problem
   The description of the problem is usually done by ourselves or someone who has it. In this step it is important to clarify all its 'dark' points and to have no doubt as to its wording.
2. Find the data.
   Finding the data means what these elements are based on in order to solve a problem, for example in an equation $ax + b = 0$ data are the factors a and b.
3. Find the requested.
   The information that we need to find to deal with the problem. Continuing the previous example information is the x of the equation $ax+b=0$.

Searching for a solution or set of solutions.

Once we have recognised the data and what is requested, a solution must be found to solve the problem. Often that's not easy. Thus, it is necessary to look for a method or **a logical set of steps leading to the solution**.

These steps must lead to a solution **whenever** the same problem arises and, moreover, be done in **a relatively short period** of time.

In mathematics and computer science, an algorithm is a finite sequence of well-defined, computer-implementable instructions, typically to solve a class of problems or to perform a computation.

Choosing the right solution

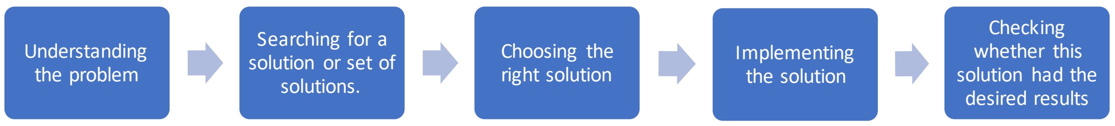
Often a problem can have more than one solution or some of them may be more efficient than others. Choosing the solution correctly leads to a shorter and more reliable outcome.

Implementing the solution

After the resolution method is selected, a series of actions must be taken to implement it. In other words, apply **the Algorithm.**

Checking whether this solution had the desired results.

Finally, after applying the solution it is necessary to test with various data, to examine whether it leads to correct results each time it is performed without problematic situations.


Understanding the problem
Searching for a solution or set of solutions.
Choosing the right solution
Implementing the solution
Checking whether this solution had the desired results

Picture 1 The steps

Lesson 3 - My first Program

⏱ **2h**

What students should know

- How to start B4J
- How to create and save a new project
- How to run a project
- What is error screen.
- How to see Turtle commands
- How to write a new project using Turtle

Hello World

Make a program that draws a straight line with the help of the turtle on the computer screen.

Recommended instructions for the instructor

The aim of the above exercise is to familiarize students in the creation of a project with B4J and to become the first acquaintance with the programming environment. The following instructions in no way limit the instructor to adapting his course to the relevant educational conditions.

Teachers tip

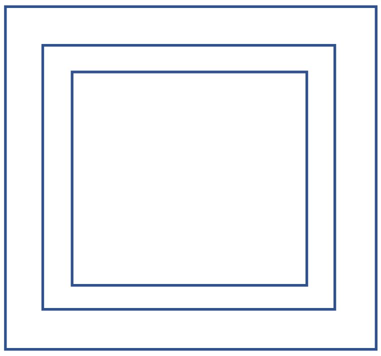
This is students first lesson. Therefore, keep it simple!

	Function	**Description**
1	How to create a first project with B4J	*Menu File -> New -> B4Xturtle* *Project Folder* *Project Name* Explain to students the importance of the right names in each project they create and the value of correct storage in folders in a structured way
2	Run Project	*Menu Project -> Compile & Run* Explain what means compile and how to recognize syntax errors in log. You don't need to provide to much information about compilation. Just the basic stuff in order to run a project.
3	#Region Project Attributes	Change Values in *MainFormWidth* and *MainFormHeight* to make different size application
4	Sub Turtle_Start	*What means Sub?* A small amount of code which is doing a certain activity.

Function	Description
5 Turtle methods	What is Turtle? What .MoveForward do? How can we find more commands? (Tell students to write "Turtle." To see the list of methods.
6 Errors	How to identify an error in log screen

Exercises

1. Using turtle and methods ***MoveForward***, ***TurnLeft***, ***TurnRight*** draw a square with any size you want.
2. Using Previous commands and ***PenUp***, ***PenDown*** and ***Move*** draw 3 squares like the image bellow.



3. Using previous commands design a sketch of your choice. Give a name to your sketch and explain how you did it.

Lesson 4 - Variables and Range

⏱ **3h**

What students should know

- Explain how a computer stores data in ram.
- Explain what a variable is.
- How to name a variable
- Assign a value to a variable.
- Use Mathematical Operators
- Use log command to display a variable

Imagine that you live on a street with a few million houses all in a row; each house has as you all know its address which starts at number 1 and ends at the last house; in order to be able to locate a friend who lives on that street you need to know the number of the house; so we have on the one hand a house number and on the other the friend who lives in that house.

Figure 1 Computer Memory (https://www.freepik.com)

The computer's central memory works in much the same way. There are many houses each with its address and a "resident" within each house. This address is called memory address and the "resident" content. On the computer often the "resident" who from now on we will call variable needs more houses to fit.

For a programmer to use memory, he must know the data he needs and the type of data. These can be integers or real numbers, words or letters, some reasonable values (truth, false). He also needs a "home" in the computer memory to store them represented by address.

In B4X the data can be stored in different types as:

B4X	**Type**
Boolean	boolean
Byte	integer 8 bits
Short	integer 16 bits
Int	integer 32 bits
Long	long integer 64 bits
Float	floating point number 32 bits
Double	double precision number 64 bits
Char	character

String	array of characters

Table 1- Simple types of variables

Every type needs different space in memory to store values.

Because it is difficult for the developer to remember all the addresses of his data, each address corresponds to a name. Fortunately, in fact this is done by the programming language itself and all it takes is to think of a good name for his data. For example, a data that is an integer for age could be called "age". Now, there is a "home" called age in computer memory.

Remember

Variables are used to store information to be referenced and manipulated in a computer program. They also provide a way of labeling data with a descriptive name, so our programs can be understood more clearly by the reader and ourselves. It is helpful to think of variables as containers that hold information.

How to find how many variables you need

In any programming problem that a developer encounters, they should be able to locate the data and the information's of the problem.

In programming we name all those elements that we need to know to move forward in solving a problem. Usually in a programming problem we find them in the pronunciation of the exercise with the help of keywords such as:

- Reads
- Registers
- Ask
- Accept
- Type

Example 1: Write a program that converts the euros we <u>type</u> into dollars.

Example 2: Make a program that <u>accepts</u> a positive integer and calculates its square, cube, and square root.

Information's

Information's in programming are all the elements that we need to calculate after processing our data. We usually find them in pronunciation using keywords such as:

- Calculates
- Displays
- Writes
- Counts
- Convert

In the previous examples what are the requested?

Naming Variables

The names of the variables in B4X must follow specific rules.

- They must start with a capital or small character.
- They can then have digits or the character underscore.
- B4X does not distinguish capital and small letters.

Also, it's a good practice to name variables beginning with 3 small letters indicating the kind of a variable and continue with 1 uppercase letter and a meaningful word. For example:

- Dim **intAge** as Int
- Dim **fltAmount** as Float
- Dim **strName** as String

This practice helps a lot when you find variables in the code to recognize the type and the value it stores.

Declaring Variables

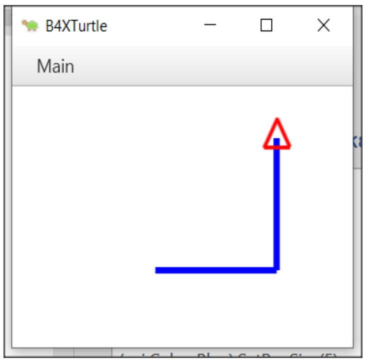
My First Variable

Make a program that assigns a value into integer and then draws with the help of the turtle a line of length as long as the value in the variable.

In B4X to use a variable we must first inform the language of its existence in order to commit space in the computer's memory to store its value.

For example, in the following code, the statement is as follows:

```
'Program: My First Variable
Sub Turtle_Start
   Private intDistance As Int
   Public intTurn As Int
   intDistance = 100
   intTurn = 90

   Turtle.SetPenColor(xui.Color_Blue).SetPenSize(5)
   Turtle.MoveForward(intDistance)
   Turtle.TurnLeft(intTurn)
   Turtle.MoveForward(intDistance)
End Sub
```



The declaration of variables begins with the keyword Private or Public.

Private means that the variable is known only in the specific space declared and no other program or subprogram does not know its existence and thus the value it contains.

Instead, a variable statement that starts with the keyword Public can be known to other programs or subprograms or classes etc.

After the keyword Private or Public follows the name of the variable we chose to give. This is where the rules discussed above apply. Finally, the type of the variable follows. For simple variables these are all those described in the *Table 1- Simple types of variables*.

Teachers tip

You don't have to explain all the variables already as well as their use. For your students to start programming, the basics of integer, float, string are enough. As you progress through the courses you can include other types according to your needs.

Comments

In computer programming, a comment is a programmer-readable explanation or annotation in the source code of a computer program. They are added with the purpose of making the source code easier for humans to understand and are generally ignored by compilers and interpreters. The syntax of comments in various programming languages varies considerably. (Wikipedia, 2021)

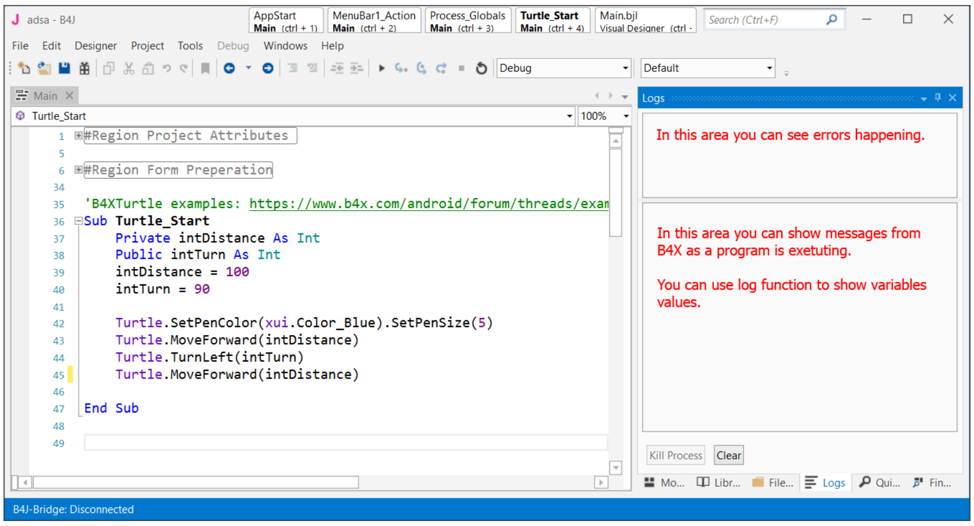
In B4X comments are inserted by writing the character ' as their first letter. From this point on it is not recognized by the translator of the language. Generally, in B4X comments you should put anywhere it is important to remember what you are doing as well as before the subprograms to explain what their job is. Comments are easily distinguished in code from the green color given to them by the programming environment (IDE).

Example

```
'Program: My First Variable
'This program draws a right angle, with sides as much as the
'value of the intDistance variable
Sub Turtle_Start
   Private intDistance As Int
   Public intTurn As Int
   intDistance = 100     'The sides of the right angle
   intTurn = 90          '90o angle

   Turtle.SetPenColor(xui.Color_Blue).SetPenSize(5)
   Turtle.MoveForward(intDistance)
   Turtle.TurnLeft(intTurn)
   Turtle.MoveForward(intDistance)
End Sub
```

The log area and the log function.

During programming various errors occur. Generally, errors in programming are divided into two categories syntax and logical. For now, we will deal with the syntax errors that are recognized by the programming language and indicate them on the logs screen. In order to access the logs screen we need to click on the relevant logs tab at the bottom right. The Logs screen itself is divided into two frames, the first of which displays errors and the bottom screen displays language messages or that information we want to display using the log() function. Using the Log() function helps the developer display messages while running a program as well as variable values to help control the program's proper operation.



```
#Region Project Attributes

#Region Form Preperation

'B4XTurtle examples: https://www.b4x.com/android/forum/threads/exam
Sub Turtle_Start
    Private intDistance As Int
    Public intTurn As Int
    intDistance = 100
    intTurn = 90

    Turtle.SetPenColor(xui.Color_Blue).SetPenSize(5)
    Turtle.MoveForward(intDistance)
    Turtle.TurnLeft(intTurn)
    Turtle.MoveForward(intDistance)

End Sub
```

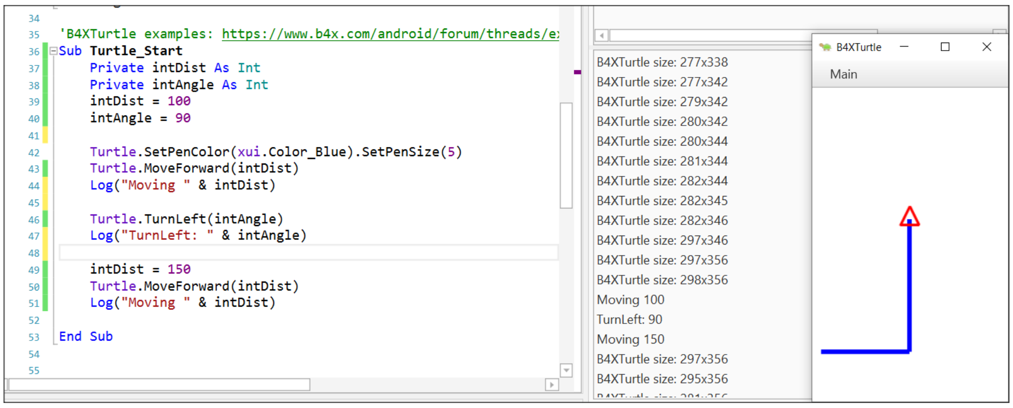
In this area you can see errors happening.

In this area you can show messages from B4X as a program is exetuting.

You can use log function to show variables values.

Figure 2 Logs Screen

To display any information on the screen it is sufficient to use the log() function as the example of the following image.



```
'B4XTurtle examples: https://www.b4x.com/android/forum/threads/e:
Sub Turtle_Start
    Private intDist As Int
    Private intAngle As Int
    intDist = 100
    intAngle = 90

    Turtle.SetPenColor(xui.Color_Blue).SetPenSize(5)
    Turtle.MoveForward(intDist)
    Log("Moving " & intDist)

    Turtle.TurnLeft(intAngle)
    Log("TurnLeft: " & intAngle)

    intDist = 150
    Turtle.MoveForward(intDist)
    Log("Moving " & intDist)

End Sub
```

```
B4XTurtle size: 277x338
B4XTurtle size: 277x342
B4XTurtle size: 279x342
B4XTurtle size: 280x342
B4XTurtle size: 280x344
B4XTurtle size: 281x344
B4XTurtle size: 282x344
B4XTurtle size: 282x345
B4XTurtle size: 282x346
B4XTurtle size: 297x346
B4XTurtle size: 297x356
B4XTurtle size: 298x356
Moving 100
TurnLeft: 90
Moving 150
B4XTurtle size: 297x356
B4XTurtle size: 295x356
```

Figure 3 Using log function

Mathematical Operators

B4X supports all known mathematical operations:

Operator	Example	Operation
+	x + y	Addition
-	x - y	Subtraction
*	x * y	Multiplication
/	x / y	Division
Mod	x Mod y	Modulo
Power	Power(x,y) x^y	Power of

Examples:

```
Private intA, intB, intC, intS As Int
Private fltD, fltM As Float
intA = 40
intB = 20
intC = 30

intS = intA + intB + intC
Log(intS)                   'Shows 90

fltD = intS / 3
Log(fltD)                   'Shows 30

intA = intAa + 1            'Increase intA by 1
Log(intA)                   'Shows 41

intS = Power(intA - 11, 2)  ' 30²
Log(intS)                   'Shows 900

fltM = intA mod 2           '41 modulo 2
Log(fltM)                   'Shows 1
```

Strings

In computer programming, a string is traditionally a sequence of characters, either as a literal constant or as variable. The latter may allow its elements to be mutated and the length changed, or it may be fixed (after creation) (Wikipedia, Wikipedia - Strings, 2021).

A string is declared like the other variables using the String statement

```
Private strName as String
```

Assigning value to a string can be done with the = symbol or by reading a value from the user (something we'll see later).

```
Private strName, strSurName as String
strName = "George"
strSurName = "Smith"
```

Also we can string together using the character &.

```
Private strName, strSurName as String
strName = "George"
strSurName = "Smith"
Private strPerson as String

strPerson = strName & " " & strSurName
log(strPerson)        ' shows George Smith in log screen

Private strName2 as String
strName2 = "John"
strName2 = strName2 & " Smith"
```

The are also a lot of functions regarding strings that makes our life easier when we are dealing with them:

CharAt(Index)	Returns the character at the given index.
CompareTo(Other)	Lexicographically compares the string with the Other string.
Contains(SearchFor)	Tests whether the string contains the given SearchFor string.
EndsWith(Suffix)	Returns True if the string ends with the given Suffix substring.
EqualsIgnoreCase(Other)	Returns True if both strings are equal ignoring their case.
Length	Returns the length, number of characters, of the string.
Replace(Target, Replacement)	Returns a new string resulting from the replacement of all the occurrences of Target with Replacement.
StartsWith(Prefix)	Returns True if this string starts with the given Prefix.
ToLowerCase	Returns a new string which is the result of lower casing this string.
ToUpperCase	Returns a new string which is the result of upper casing this string.
Trim	Returns a copy of the original string without any leading or trailing white spaces.

Table 2 String Functions (https://www.b4x.com/android/documentation.html)

Teachers tip

You can find more information about string manipulation in language booklets at (https://www.b4x.com/android/documentation.html)

Exercises

1. In the following exercises, identify the variables you need to declare. For each of them, write the relevant statement and give it an appropriate name.
   - Calculate the volume of a cylinder with a radius of one metre and a height of two metres.
   - Make a program that accepts a positive integer and calculates its square, cube, and square root.
   - Make a program that reads a sum of money in € and calculates and displays the corresponding amount in $.
   - Write a program that reads the length of the sides of a rectangle from the keyboard and calculates and displays its area.
   - The total resistance R of two resistances R1 and R2 connected in series is R1 + R2 and parallel (R1*R2)/(R1+R2)

respectively. Male a program that it reads two values of resistant R1 and R2 and calculates the total resistance in series and parallel.

2. In the following variable names, select which are correct and which are not:
intAge

Name	True	False
int Age	☐	☐
_fltAmount	☐	☐
strName	☐	☐
1myAge	☐	☐
int_value	☐	☐

3. It's the end of the semester and you got your grades from three classes: Geometry, Algebra, and Physics. Create a program that: gives in 3 variables the grades of these 3 classes (Grades range from 0 - 10) Calculate the average of your grades.

4. You have bought a Bitcoin and now it's on the rise!!! Create a program that:
   - Assign the value of the bitcoin at the time of purchase.
   - Assign the percentage of increase (or decrease)
   - Logs the total value of your bitcoin.
   - Logs the increase or decrease value.

5. You now own some property, and you want to calculate the total area of the property. Create a program that:
   - Assign the width and height in two variables.
   - Calculate and log the area.

6. You are interested in buying a new laptop. You check the price and you see that the price is 300$ without the 10% tax. Create a program that:
   - Assign the the price of the laptop in a variable.
   - Assign the tax percentage in a second variable.
   - Calculate and logs the total amount.

7. In a company the monthly salary of an employee is calculated by the minimum wage 400$ per month, plus 20$ multiplied by the number of years employed, plus 30$ for each child they have. Create a program that:
   - Assign the number of years employed in a variable
   - Assign the number of children the employee has in second variable.

- Calculate and logs the total amount of salary the employee makes.

8. Create a program that log the last digit of a given integer.
9. Create two variables a and b, and initially set them each to a different number. Write a program that swaps both values.

   Example: a = 10, b = 20

   Output: a = 20, b = 10

10. Create two variables ‘a’ and ‘b’, and initially set them each to a different number. Write a program that double the Value of ‘a’ variable and increase the value of ‘b’ by 1.

    Example: a = 10, b = 20

    Output: a = 20, b = 21

Lesson 5 – Designer

⏱ 2h

What students should know

- Talking about Designer
- Design the first Screen
- Inserting and customizing Views: Labels, TextFields, Buttons, Panes
- Saving forms
- Design their own Main Screen using wireframes

Until now you have used the turtle to move it through the screen, and the log command to display information on the language log screen. What if you ask the program user to enter values? Or what happens when you want to display information to the user? The B4X has a special interface screen design environment. Through it you can design the appearance of the screens and generally communicate with the users of your application.

Every time you must design an app you should keep in mind that the look of your app is what will attract users to it. In other words, it is not enough to be simple functional but also easy to use as well as to offer information in an organized way without confusing.

Before designing any app remember some key design elements (usabilty.org, 2021):

Keep the interface simple. The best interfaces are almost invisible to the user. They avoid unnecessary elements and are clear in the language they use on labels and in messaging.

Create consistency and use common UI elements. By using common elements in your UI, users feel more comfortable and can get things done more quickly.

Strategically use color and texture. You can direct attention toward or redirect attention away from items using color, light, contrast, and texture to your advantage.

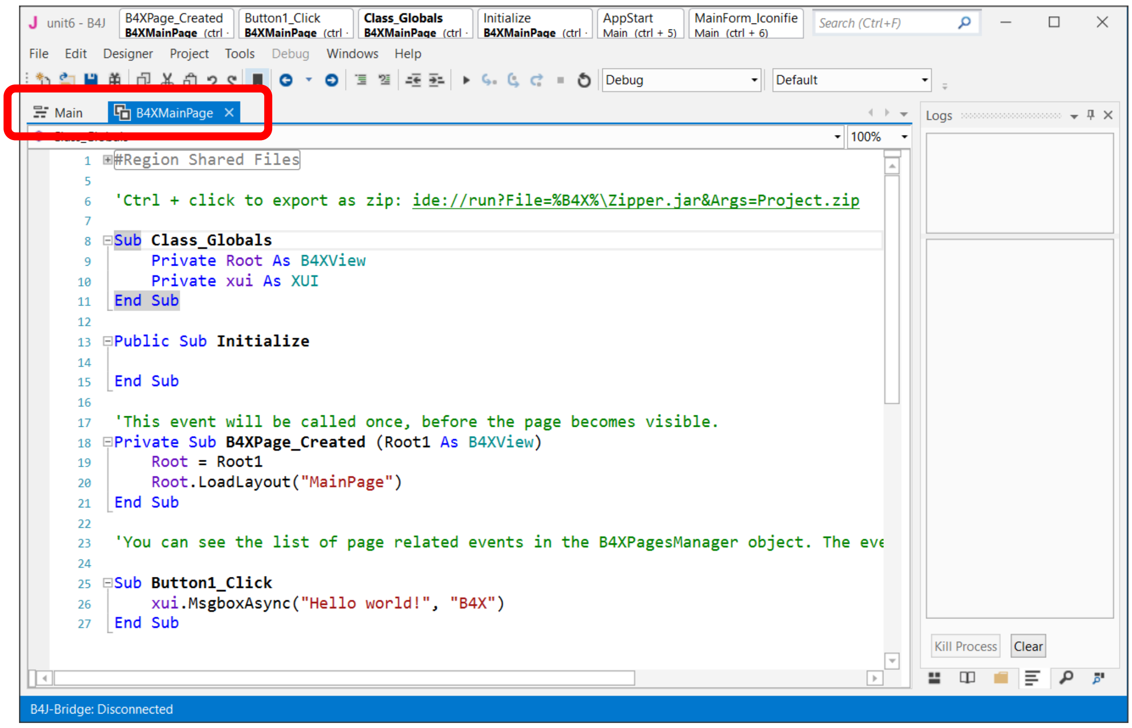
Use typography to create hierarchy and clarity. Carefully consider how you use typeface. Different sizes, fonts, and arrangement of the text to help increase scanability, legibility and readability.

Make sure that the system communicates what's happening. Always inform your users of location, actions, changes in state, or errors.

Think about the defaults. By carefully thinking about and anticipating the goals people bring to your site, you can create defaults that reduce the burden on the user. This becomes particularly important when it comes to form design where you might have an opportunity to have some fields pre-chosen or filled out.

First steps with design

First of all, you should begin the B4J and now from file menu choose **New** and **B4XPages.** Choose a directory and write a name for your project. You will see the code bellow. There are two tabs of code here, the first one called **Main** and the second **B4XMainPage**.

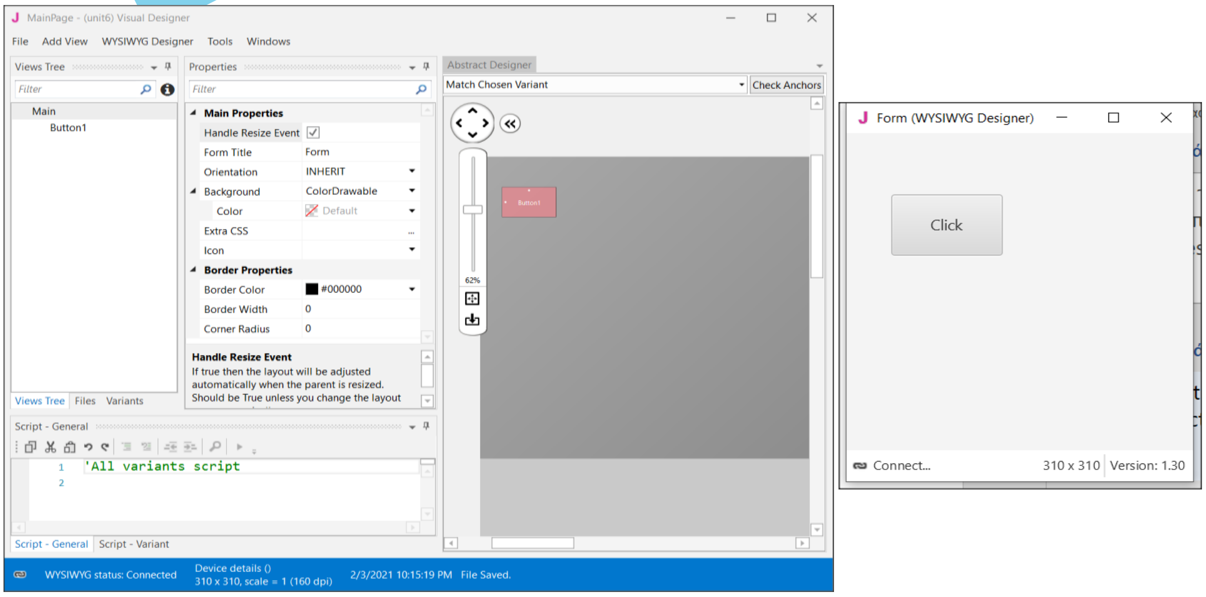

unit6 - B4J
B4XPage_Created B4XMainPage (ctrl
Button1_Click B4XMainPage (ctrl
Class_Globals B4XMainPage (ctrl
Initialize B4XMainPage (ctrl
AppStart Main (ctrl + 5)
MainForm_Iconifie Main (ctrl + 6)
Search (Ctrl+F)
File Edit Designer Project Tools Debug Windows Help
Debug
Default
Main
B4XMainPage
100%
Logs
Kill Process
Clear
B4J-Bridge: Disconnected

```
#Region Shared Files

'Ctrl + click to export as zip: ide://run?File=%B4X%\Zipper.jar&Args=Project.zip

Sub Class_Globals
    Private Root As B4XView
    Private xui As XUI
End Sub

Public Sub Initialize

End Sub

'This event will be called once, before the page becomes visible.
Private Sub B4XPage_Created (Root1 As B4XView)
    Root = Root1
    Root.LoadLayout("MainPage")
End Sub

'You can see the list of page related events in the B4XPagesManager object. The eve

Sub Button1_Click
    xui.MsgboxAsync("Hello world!", "B4X")
End Sub
```

Do not worry about them now. We will discus later about them. Now all you need to know is that inside the B4XMainPage all the beautiful things happen for our code!

Now from the **Designer menu** select **Open Internal Designer.**

This is where the design process begins. Two windows will open, the first is the designer and the second is the preview of the screen you are designing.



Main Properties	
Handle Resize Event	✓
Form Title	Form
Orientation	INHERIT
Background	ColorDrawable
Color	Default
Extra CSS	
Icon	
Border Properties	
Border Color	#000000
Border Width	0
Corner Radius	0

Handle Resize Event
If true then the layout will be adjusted automatically when the parent is resized. Should be True unless you change the layout

```
'All variants script
```

WYSIWYG status: Connected Device details () 310 x 310, scale = 1 (160 dpi) 2/3/2021 10:15:19 PM File Saved.

Figure 4 Designer Screen

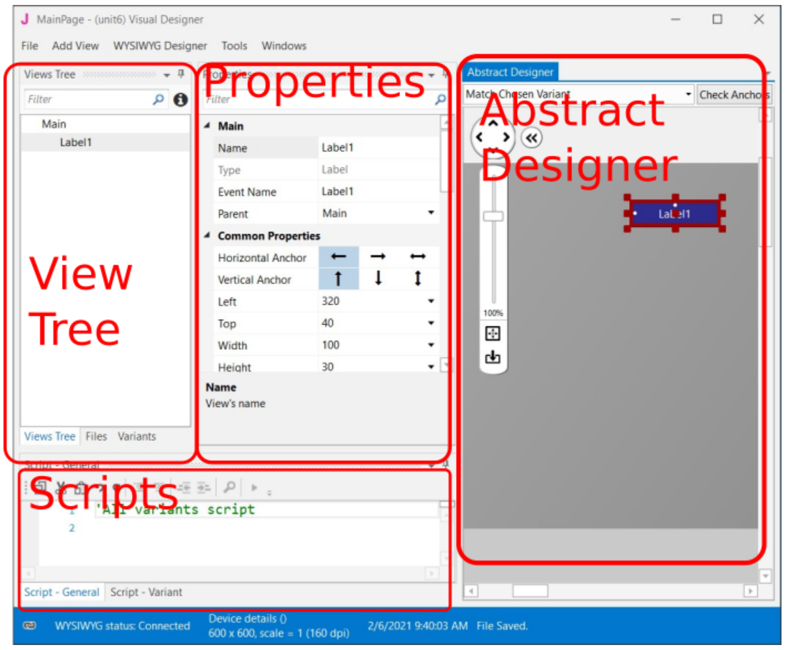
Visual designer

The AddView menu includes all the objects needed to create our screen.

Select a label from the menu, and then move it to the preview screen where you decided before in the wireframe stage.

Remember

You can move all objects around the view by selecting them and holding the left mouse key.



Main	
Name	Label1
Type	Label
Event Name	Label1
Parent	Main
Common Properties	
Horizontal Anchor	
Vertical Anchor	
Left	320
Top	40
Width	100
Height	30

Name
View's name

```
'All variants script
```

WYSIWYG status: Connected Device details () 600 x 600, scale = 1 (160 dpi) 2/6/2021 9:40:03 AM File Saved.

Figure 5 Designer's parts

The Views Tree
Here you see all the objects in your design. Keep in mind that the objects above the list are placed behind the next ones.

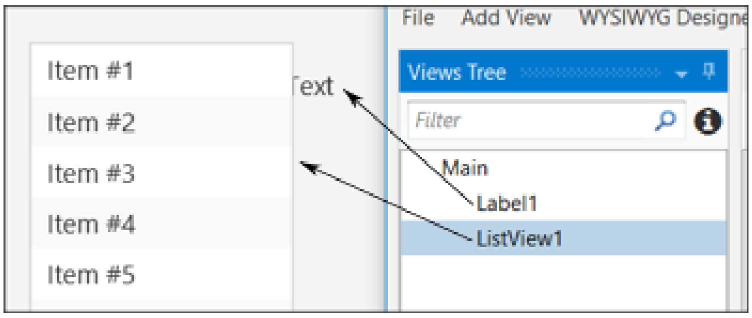


Figure 6 Views Tree

Properties
Each object has properties such as size, screen position, colors, font, etc. Each property can be changed either through the properties option or later through the program code.

One of the most important properties is the name of the object. This, like variables, should follow specific rules to indicate their type. For example in *Table 3* Naming objects some cases of names.

Type	**Prefix**	**Example**
Label	lbl	lblName
Button	btn	btnSave
TextField	txt	txtAge
Spinner	spn	spnYears
Pane	pn	pnLine1

Table 3 Naming objects

Abstract Designer
The Abstract Designer allows to select position and resize Views. It is a very useful function for quickly placing objects in the correct position (however the most accurate placement is made by the Properties tab by setting the relevant values).

Example 1
Imagine that you want to make a program that reads from the screen two Integers, calculates, and shows the sum.

Decide on the size of the app screen.
This depends on the amount of information we must display as well as on the individual items such as menus, graphics etc.

To set the application's size before beginning the Designer first go to **Main** Tab and change the first lines of code Width and Height:

```
#Region Project Attributes
    #MainFormWidth: 600
    #MainFormHeight: 400
#End Region
```

Save your project and open Designer.

Set an appropriate variant.

Usually, you should set the variant as the MainFormWidth and MainFormHeight. This will help you plan without the risk of going outside the screen limits.
Choose Variants and then New Variant and set the width and height.
You can have as many variants as you want for different screen size but for now, we stay to only one.
Also, you can remove any variant by selecting it and choosing "Remove Variant".

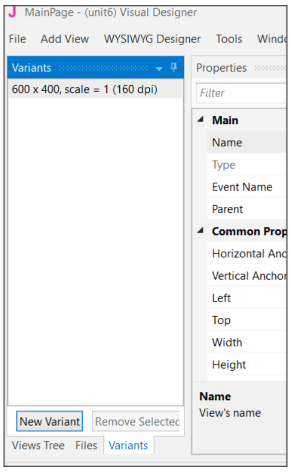


Figure 7 Variants Screen

Design a wireframe.

For small applications, this step is optional, but it is a good habit to have decided from the beginning where you want to display your details. You can use a simple sheet of paper or several programs to help create previews.

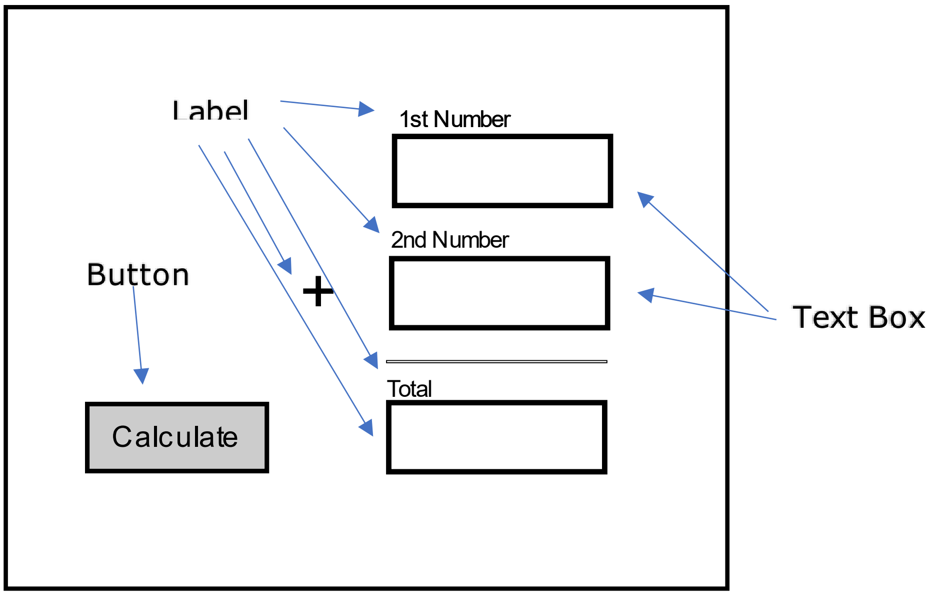


Figure 8 Wireframe

Create the views.

Now that you know what you need and where to place it, use the Designer tools to complete the process.

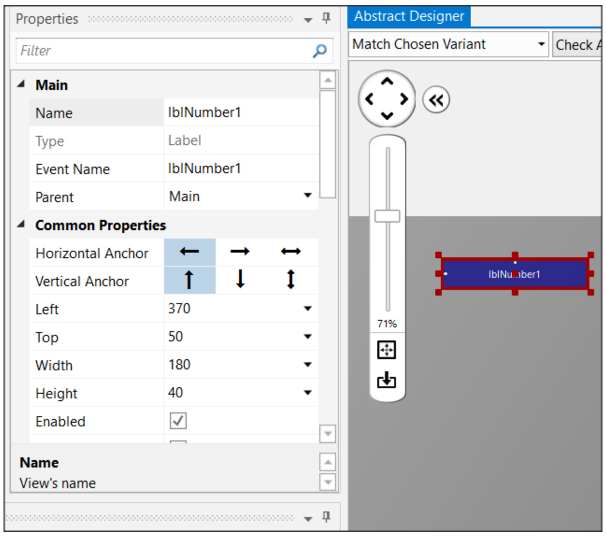
Inserting a label.

From View menu select label and you will see a label object in your View Tree and in the Abstract Designer. Move it in the place you decide in wireframing and choose an appropriate name from properties.

Now scroll down the Properties and set:

- **Width**: 180
- **Height**: 30
- **Text**: First Number
- **Allignment**: CENTER_LEFT
- **Font**: SansSerif
- **Size**: 13

Experiment with the other settings and see it displayed in the preview pane.



Properties

Main	
Name	lblNumber1
Type	Label
Event Name	lblNumber1
Parent	Main
Common Properties	
Horizontal Anchor	
Vertical Anchor	
Left	370
Top	50
Width	180
Height	40
Enabled	✓

Name
View's name

Figure 9 Labels

Insert a second label or you can also dublicate the first one. Select it and press Ctrl-D. The second method gives a same label as the first one with the same properties except Name Property. Set "lblNumber2" as name and "Second Number" as Text and Create a third label with name "lblTotal" and Text: "Total".

Inserting a Text Field.

Text Fields are used to import data into the program. There is no restriction on the type of data you can import. From View Menu choose TextField and set:

- **Name**: txtNumber1
- **Width**: 180
- **Height**: 40
- **Font**: SansSerif
- **Bold**: checked

Place the textField underneath "First Number" label. Now put a same TextFIeld with name "txtNumber2" and put it underneath label "Second Number". At the end create a third TextField with Name "txtTotal". You will probably see something like the Figure 10 Text Fields

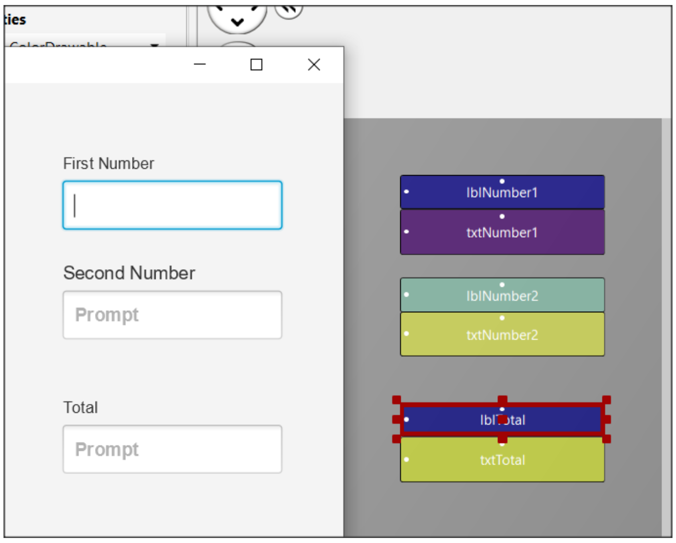


Figure 10 Text Fields

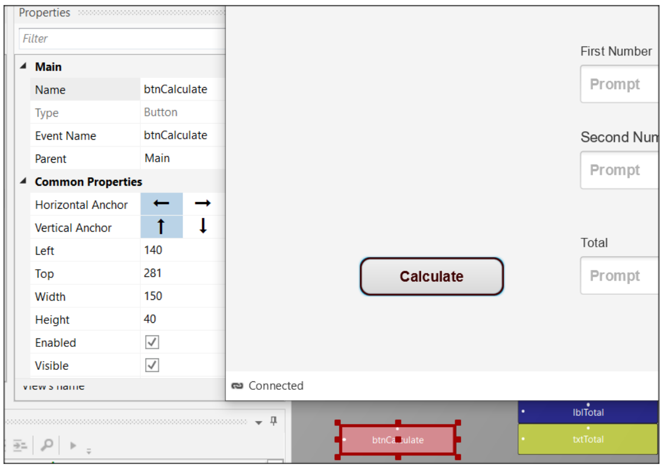
Inserting a button

Buttons in an app are used to enable functions. The program detects the click and then executes appropriate commands depending on the button pressed.

For each button you can set different features such as size, color, shape, etc. to stand out on your screen and be easily detected by users of your app.

From the Views menu select Button, and then set:

- **Name**: btnCalculate
- **Width**: 150
- **Height**: 40
- **Border Color:** #3C0000
- **Border Width**: 2
- **Corner Radius:** 20
- **Text:** Calculate
- **Text Color:** #FF3C0000
- **Font**: SansSerif
- **Size**: 15
- **Bold**: checked



Properties	
Main	
Name	btnCalculate
Type	Button
Event Name	btnCalculate
Parent	Main
Common Properties	
Horizontal Anchor	← →
Vertical Anchor	↑ ↓
Left	140
Top	281
Width	150
Height	40
Enabled	✓
Visible	✓

Figure 11 Buttons

Remember

File -> Save (or Ctrl – S) every time you make something valuable!

Inserting a Pane

You can use a Pane to visually group specific objects on the screen you're drawing. The pane displays a frame, and you can specify properties such as color, border, fill, etc. You can also use it at a very small height (1 or 2) to display a single line on your screen.

This example is used to draw a line before the total. From menu AddView insert Pane and set:

- **Name**: pnLine
- **Width**: 180
- **Height**: 1
- **Border Color:** #000000
- **Border Width**: 2

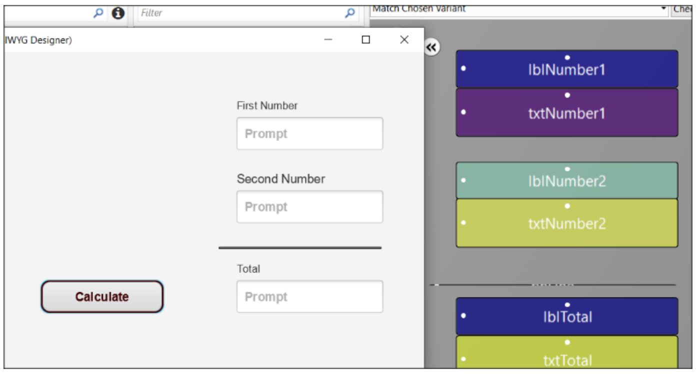


Figure 12 Pane

Now from menu File save the form. The form has already a name (MainPage) so you don't need to give an other name.

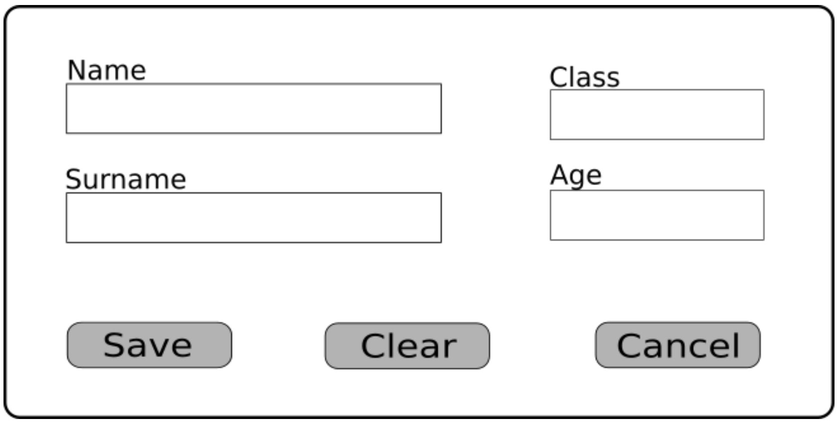
Anywhere Software

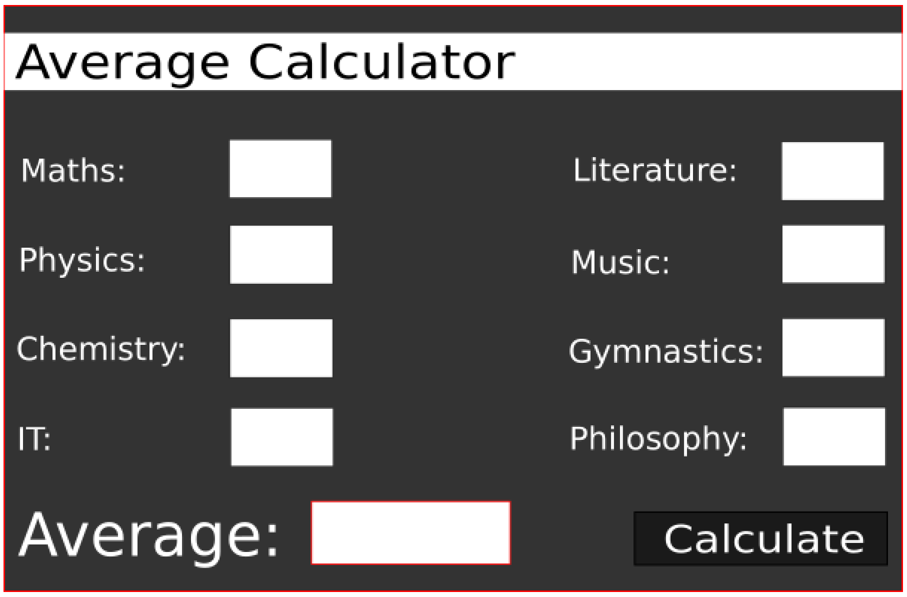
Excercises

1. Use the designer to create the following wireframes.

Teachers tip

This is the fun part. You can also leave students free to experiment with views.





2. Think and design your own Dream Application. Give it a name, create a wireframe in your notebook and create the Design View.

Bibliography

Cambridge, D. o. (2021, 1 28). *Cambridge Dictionary*. Ανάκτηση από https://dictionary.cambridge.org/dictionary/english/problem

Computing At School Working Group. (2012, March). Computer for Science: A curriculum for Schools. Ανάκτηση January 20, 2021, από http://www.computingatschool.org.uk

usabilty.org. (2021, 2 4). *User Interface Design Basics*. Ανάκτηση από usabilty.org: https://www.usability.gov/what-and-why/user-interface-design.html

Wikipedia. (2021, 2 1). *Wikipedia - Strings*. Ανάκτηση από Wikipedia - Strings: https://en.wikipedia.org/wiki/String_(computer_science)

Wikipedia. (2021, 1 31). *wikipedia.org*. Ανάκτηση από https://en.wikipedia.org/wiki/Comment_(computer_programming)