RUNEAT TUTORIAL

RunEat is an application developed by AppInventor 2.0, an open source software developed by MIT.
The application aims to help people to have a healthy diet. This is done by calculating the calories, BMI and BMR, providing the pedometer, a device for calculation of the steps taken during the day and that, combined with the section of the calculation and curiosity advises users what to eat and what activities it is better to play in maintaining a healthy body.
The application is suitable for people aged between 10 and 99 years thanks to its simple and intuitive interface.

RUNEAT DESIGN

The design of the application is very simple. We created simple images with the Paint.net program.
Each image was conceived by a group of designers who were concerned only with the graphic part of the application.
Each image recalls the combination of white / orange colors we chose to associate with our application.
The orange used is FF9800.

STEP BY STEP

1. HOME

Figure 1

The Application Home has the logo in the top center of the screen, followed by four buttons pointing to the different parts of the application:
- A flame for calories
- One foot for the pedometer
- A man for exercises
- A chef for recipes

Each button, when pressed, will allow you to open a new screen with the dedicated application part.

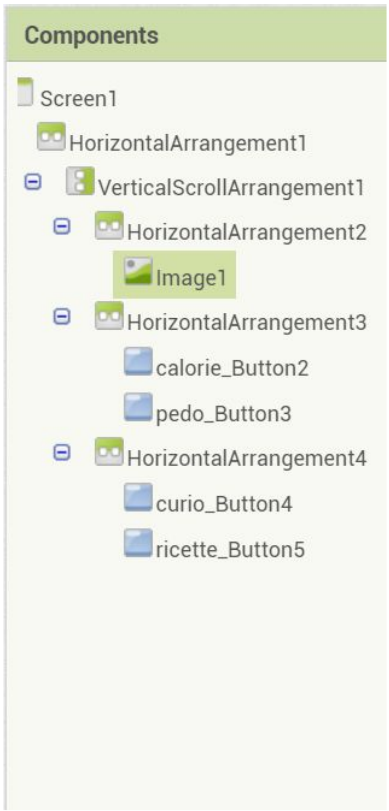
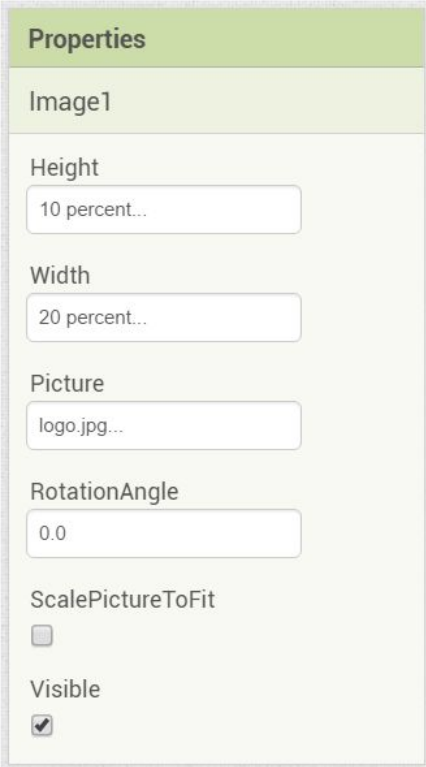


Figure 2

So let's go with the components of our app and in particular an HorizontalArrangement that we find in the Layout menu.
Subsequently we will insert a VerticalScrollArrangement within this box, which will serve if we have smaller screens of the page size and three HorizontalArrangement.



The first HorizontalArrangement will have as its property:

- AlignHorizontal: center: 3
- AlignVertical: top: 1
- Height: 10%
- Width: fill parent

In this box we will insert the image of the application logo, loading it from the properties that are on the right side of AppInventor. Always include in the properties as height and width data respectively 10% and 20% and click on the box that makes this image visible.

Figure 3

Let's now go to the second HorizontalArrangement that will contain the first two buttons, the one related to the calories and the one related to the pedometer. The procedure for button insertion is very simple, just click on User Interface and then on Botton. Rename each button so it will be easier to write the code later. To rename, just select the button inside the components, and click the Rename button at the bottom of the Components pane.
In addition, each button must be depicted with the image of its function, so let's load the corresponding image into each property.
As far as the Height property is concerned, this will be a fill parent, and by the width we will have to indicate 50% so that each button will occupy half of the box.

We will do the same for the third HorizontalArrangement, but changing the button images and names to the latter.
The end result must be as in Figure 1.
Before we pass code writing, we create four more screens that we will use for the four parts of the app. We give each Screen a name.

BLOCKS:

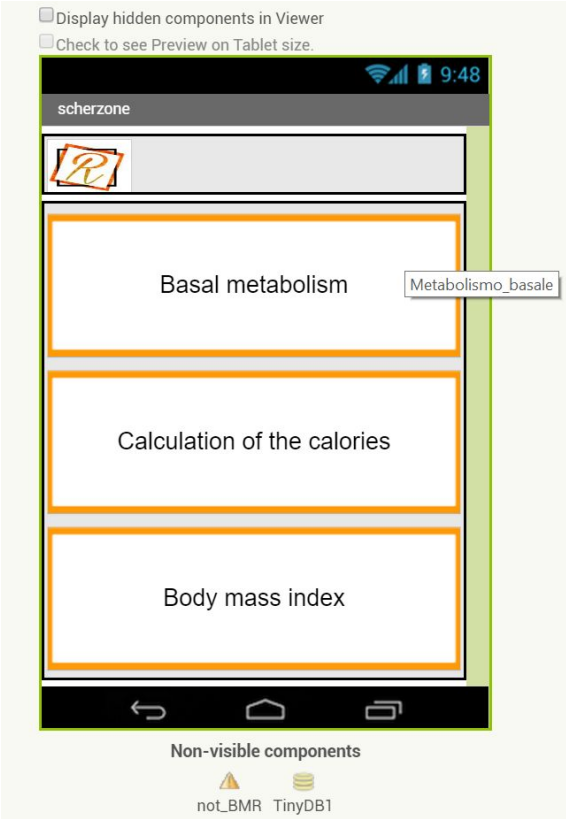
As for the code this is very simple. Let's go to the Blocks mode.
The code will cover the four buttons that each will have to open a different screen.
But the code will be the same for everyone. We put the first block of calorie brass, which we find by clicking on the calorie lock, "When calorie_Button2.Click do". This will allow us to say that when the button is clicked it will have to do an operation. Below we insert a second "open another screen screenname" check block followed by a blank text block where we will insert the screen name that will have to open that button. In my case "jokes".
This must be done for all four buttons.
Here is the code:

Figure 4

At this point Home of our application is over. We can then try out whether it works through the emulator or using our smartphone.

2. CALORIE (Screen Scherzone)

Let's now create the next screens for our application. The first we will build will be the one related to Calories.
First, let's enter a HorizontalArrangement where we put a button. In the property of the latter, we will insert the image of the application logo.
We then insert two Labels that we call met_bas and IMC respectively and in their properties we remove the flag from visible.
Next we will insert a VerticalScrollArrangement, renaming it card1, inside which we will put three buttons that we will call respectively:
Basal Metabolism
Calculation of the calories
Body mass index
Each button will look like a white button with orange border that we created earlier.
We now insert a textbox named BMR_dom and in the property we remove the visible flag.
We now insert two VerticalArrangement, which we rename met_basale and IMC_schermo respectively. In the first, we insert a textbox called IMC_dom. In the second, however, we include a Notifier and a Storage TinyDB.



At this point the graphic part of the second screen is completed, then we can proceed with the code.
The result of the graphics will be like the picture here to the side.

Figure 5

BLOCKS:

We put a checkbox "When ... click Do" and select BMR? In empty space. After DO we will connect the blocks for met_basale and BMR_dom. This block will be "set ... visible to" for both, followed by the Logical False block for the met_basale and true for BMR_dom.
This code sequence is used to hide or display components on the screen.

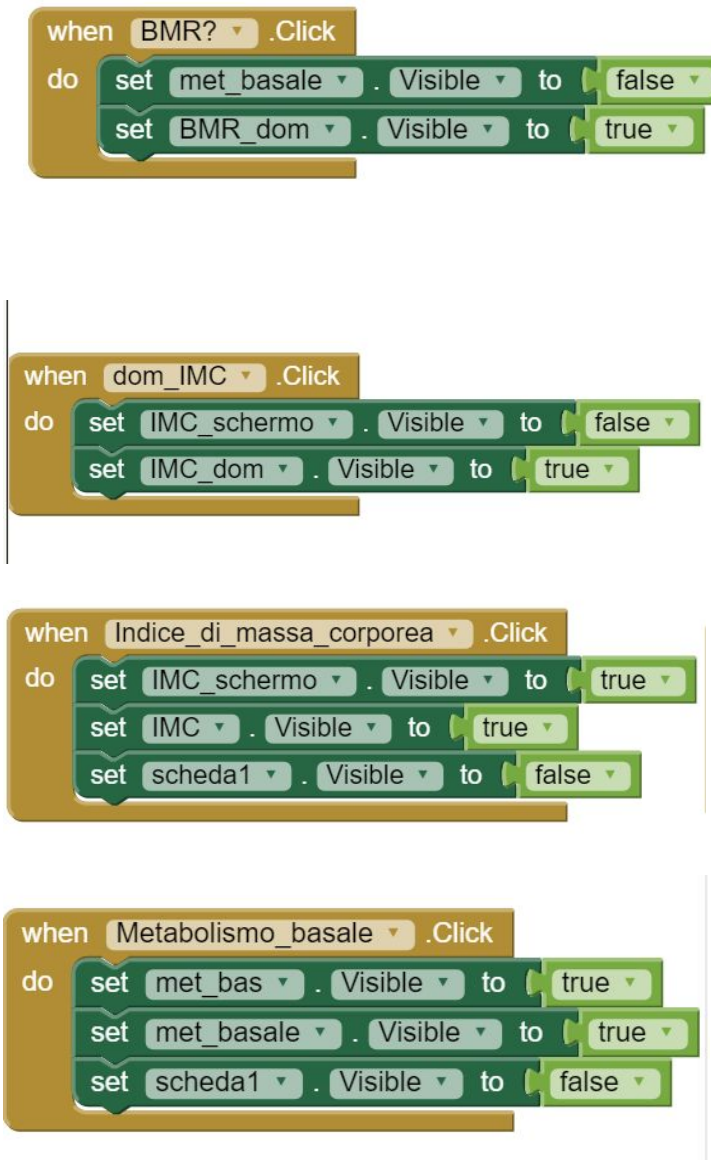


Figure 6

Let's now write the code for calculating the BMR. The code will first allow us a division based on sex, weight and age.
Below are the blocks to insert:

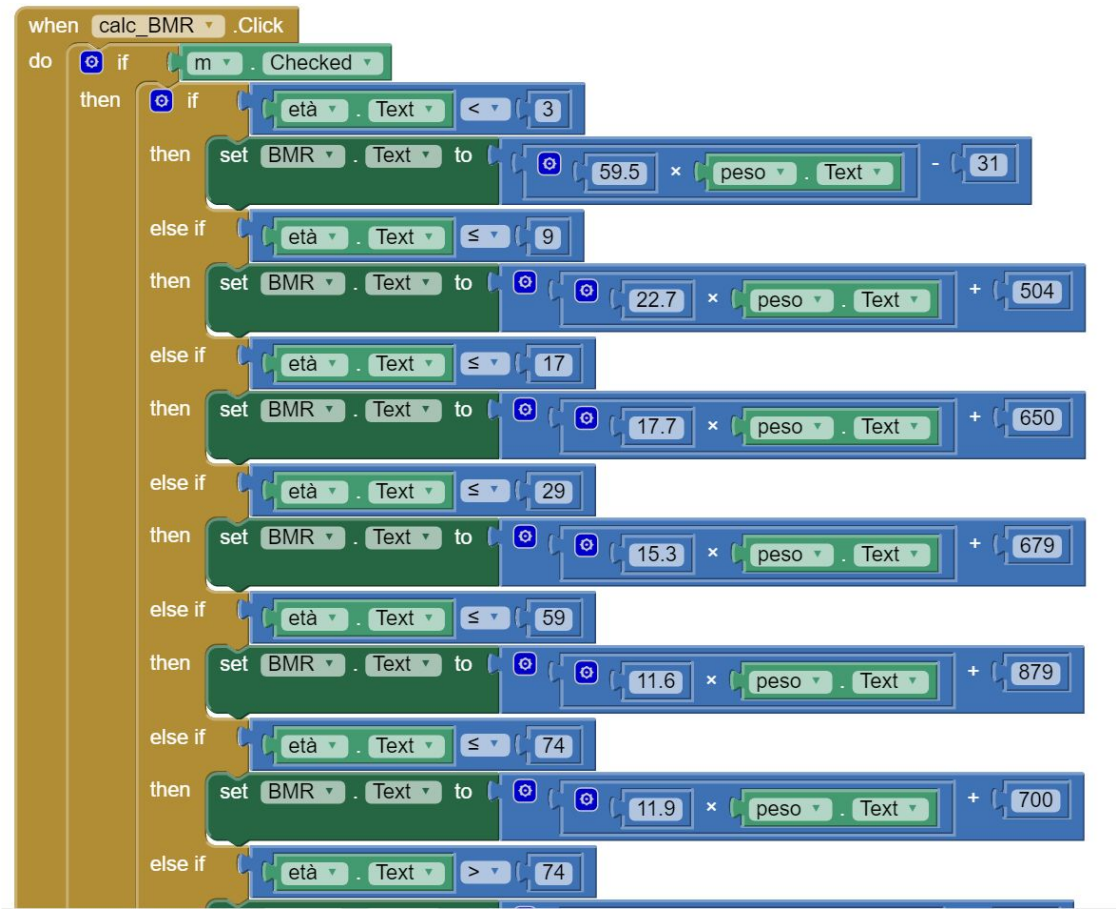


Figure 7

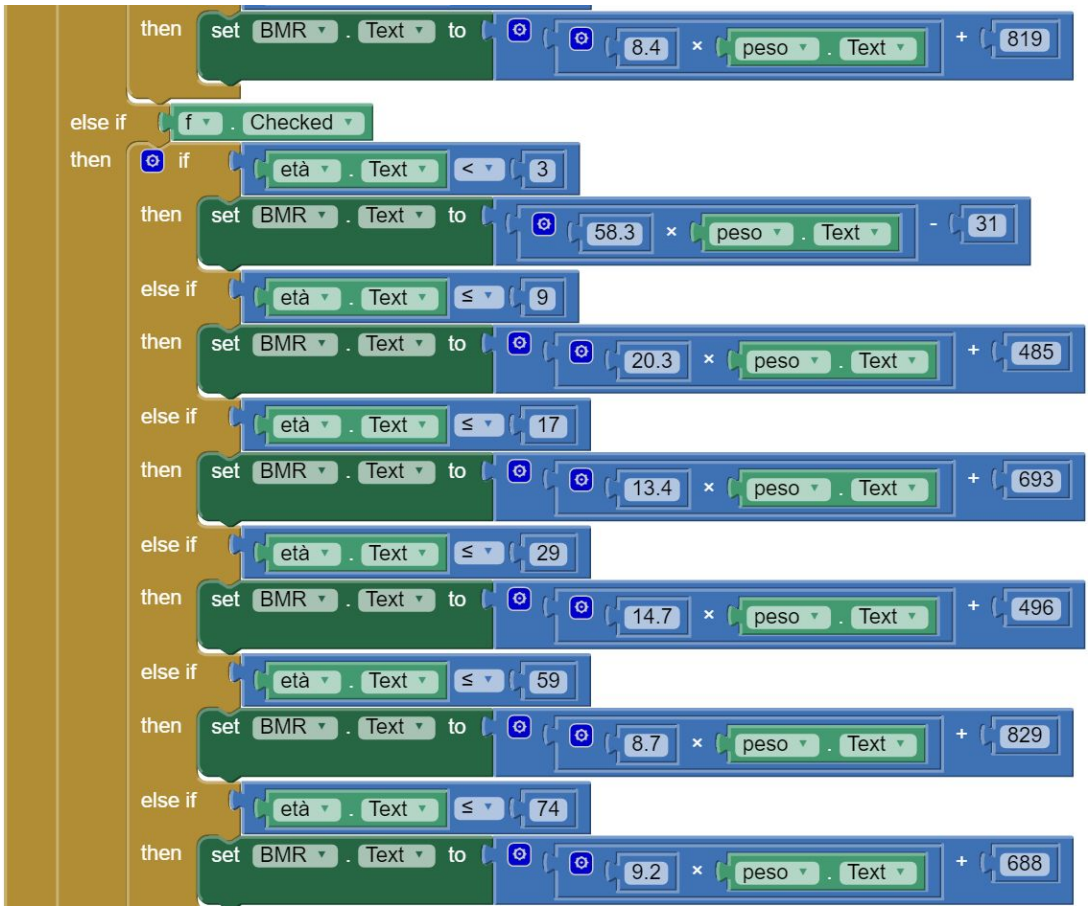


Figure 7bis

Each of these blocks calculates the BMR based on the age and weight of the subject that is using the application.

Next we will write the code for the Calculation of the Body Mass Index. The body mass index as the basal metabolism is an operation that uses the data that the user inserts. Additionally, this feature allows the user to identify his physical shape.
Below is the code that allows us to perform these operations.

Figure 8

BM and Calorie Calculation use a TinyDB data storage call to save the results, which are displayed in the archive of the day. This tool saves values by associating them with a key.
To do this we need to write code that will allow us to save the data in our database. In particular, we will have to insert these blocks:

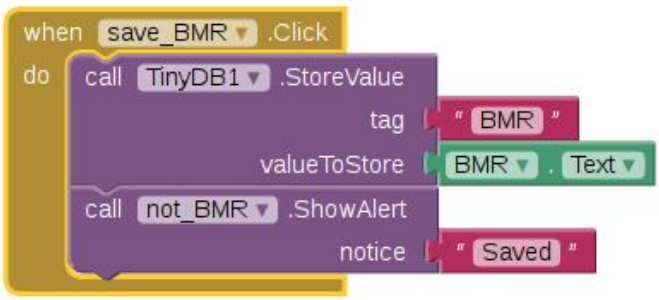


Figure 9

The next blocks to insert are those that allow us to open a new Screen when the Calculation Calorie button is pressed and return to Home when we press the logo.

Figure 10

Lastly, we need to include a code block that allows us to change the colors in our application.

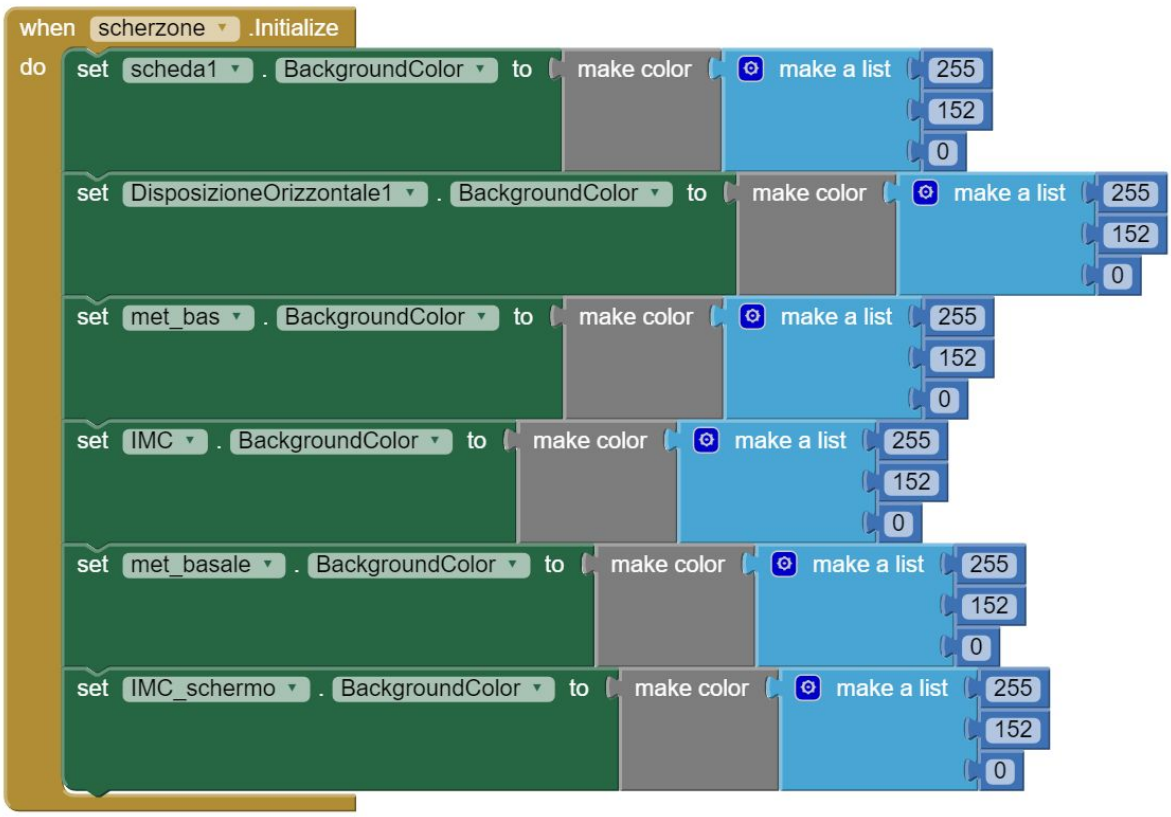


Figure 11

3. CONSUMO CALORIE

Clicking on the second button on the previous screen opens the page about calorie consumption.
By putting your weight on it, the application will indicate how many calories are consumed by performing a certain sport activity.

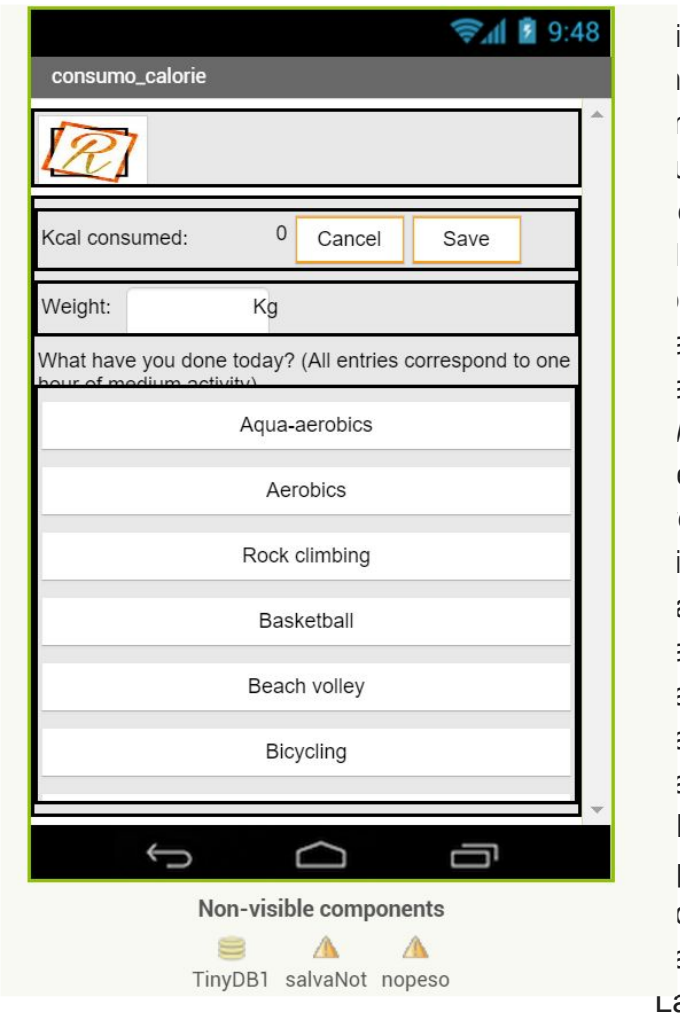


Figure 12

irst, we will create the box containing
e application logo that will allow us,
nce clicked, to return to the home of
ur application. We then insert a
orizontalArrangement in which we
ace a button with the image of the
go and two labels: one that we will
all a summary and one that will be
alled cons_cal1.
/e then insert a
erticalScrollArrangely within which
e will put other elements.
irst of these is a nine
ableArrangement label that we will
all respectively:
abel6
abel7
abel8
MR
port
ood
abel12
Label13
Label14

Subsequently, we will insert an OrizontalArrangement with four label labels15 and a tot, label16 and label17. We will also insert a button.
We now insert a VerticalArrangement called key_princ where we put three buttons: inlay_cal, cons_cal, and summary_giorn.
Now we have a VerticalArrangement with a TableArrangement inside it. In this we will put two labels, one label etichett2 and one calorie, and two bottles call and save. These will appear under their application.

Below we add another TableArrangement with a textbox that we rename the weight and three labels we call label3, label4 and label5.
We now put a last VerticalArrange where we put as many buttons as the sports we want to insert into our application. Each button will then match a sport.
The last step is the insertion of the TinyDB1 storage and two notifiers, which we find in the user interface, called saveNot and nopeso. We will then see in the code what we will use for these notifications.
At this point the screen is ready and we can switch to writing the code.

BLOCKS:

There are code blocks for each activity listed on the list.

Figure 13

The block allows you to multiply the weight placed in the Weight ... .kg field with average calorie consumption per kg.
The same block must be repeated for each sport.

We put a checkbox "When ... click Do" and select cons_cal in empty space. After the DO we will connect the blocks shown in the picture. This block will be "set ... visible to" for both, followed by Logic False and True block.
This code sequence is used to hide or display components on the screen.

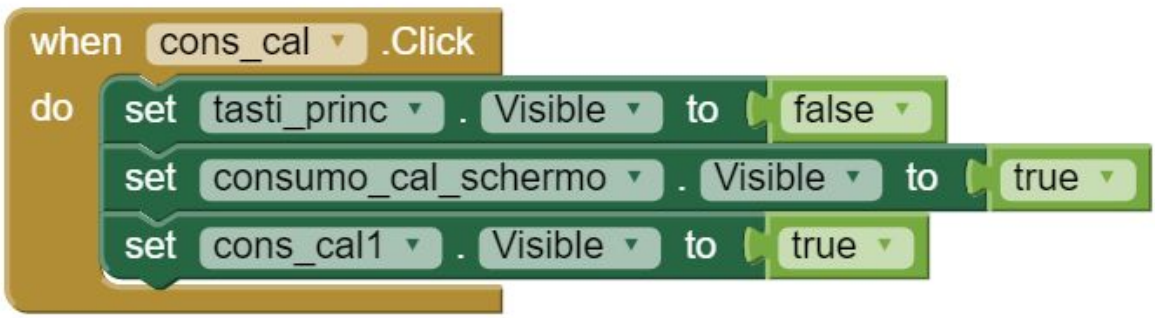


Figure 14

The block below allows us to set the calorie label text to zero at the push of the cancel button.

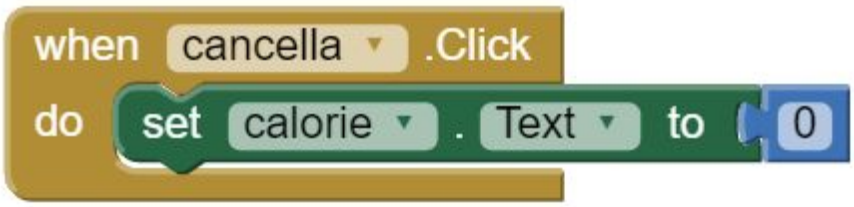


Figure 15

This check block allows you to handle errors that may occur within the Calorie Consumption page.
When an error occurs, an alert with the "Insert Weight" message is displayed.

Figure 16

The block below allows us to press the intake button to open the calorie intake screen.

Figure 17

The block below allows us to press the home button (logo) to open the home screen of our application.

Figure 18

This code sequence is used to hide or display components on the screen.

Figure 19

The following block allows you to reset the values in the database and the values in the labels indicated.

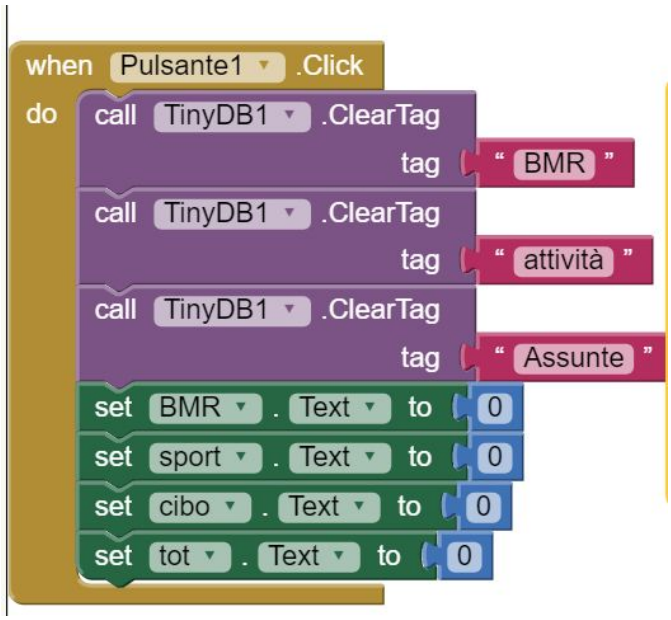


Figure 20

This block allows you to call the database and save the data inside it.

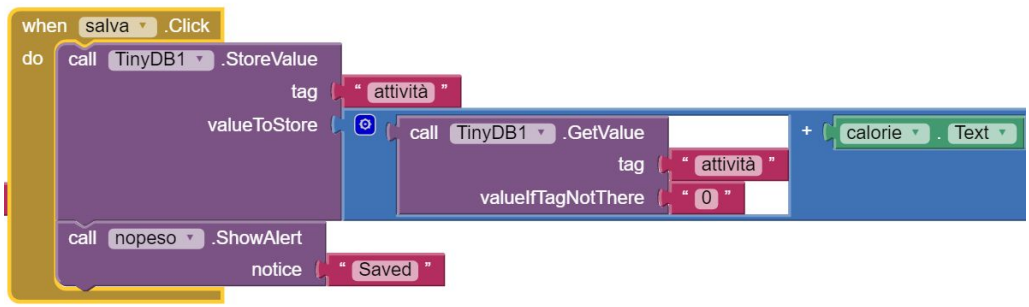


Figure 21

This block reads the data in the database and displays them on the screen.

Figure 22

Lastly, we need to include a code block that allows us to change the colors in our application.

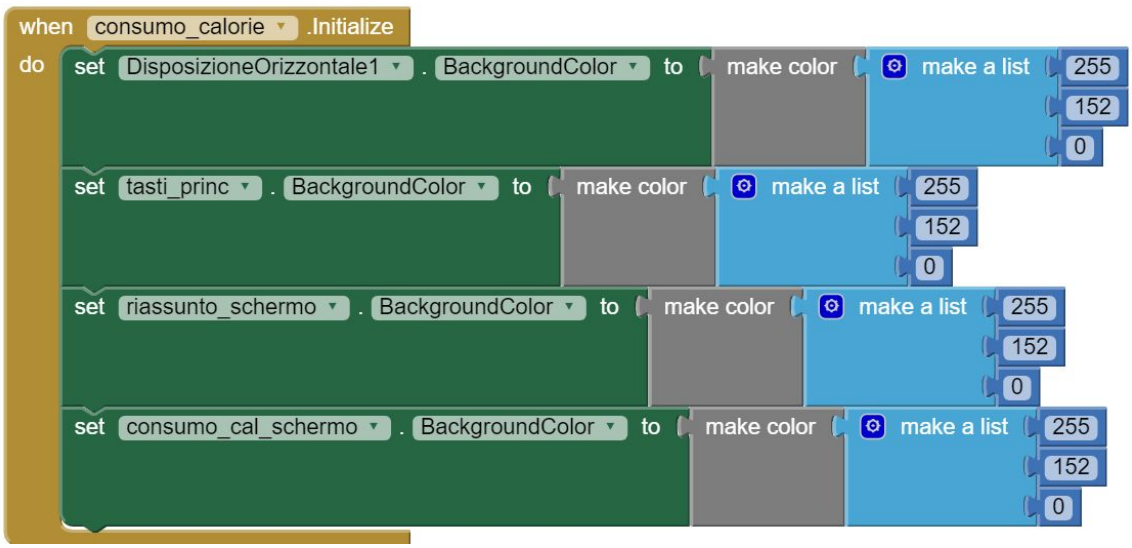


Figure 23

4. ASSUNZIONE CALORIE

This screen allows us to calculate the calories consumed over the course of the day.

Figure 23

First, we will create the box containing the application logo that will allow us, once clicked, to return to the home of our application. We then insert a HorizontalArrangement in which we place two buttons with the image of the logo and a label that we will call visible.
We now insert a OrizontalArrangement with three labels and two buttons. Labels are renamed label3, label4 and total respectively. As the bottoms we call them delete and save.
Next we insert a VerticalArrangement called addTu where we will insert a renamed label label6. Within this arrangement, we include a TableArrangement with a textboc and a label labeled label5 and a button named add.
Below we will include 4 VerticalArrangement named:

- SnackTab
- Lunch Dinner
- BreakfastTab

- Meals

Within each arrangement insert as many buttons as many as will be the foods we want to insert. In the meals we will instead put five buttons called:

- Snack
- Breakfast
- Lunch
- Dinner
- Add_tu

Finally we will insert a TinyDB and a Notifier.

BLOCKS:

The following block allows us to set the text of the label indicated to the sum of calories previously calculated plus the calories contained in the food, at the pressure of the button indicated.
The same block must be repeated for each food.

Figure 24

However, unlike those above, it allows us to enter a food that is not present in the list above.

Figure 25

The block below allows us to press the home button (logo) to open the home screen of our application.

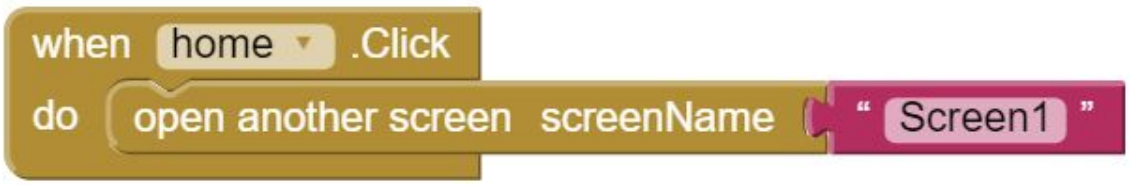


Figure 26

This block allows you to call the database and save the data inside it.

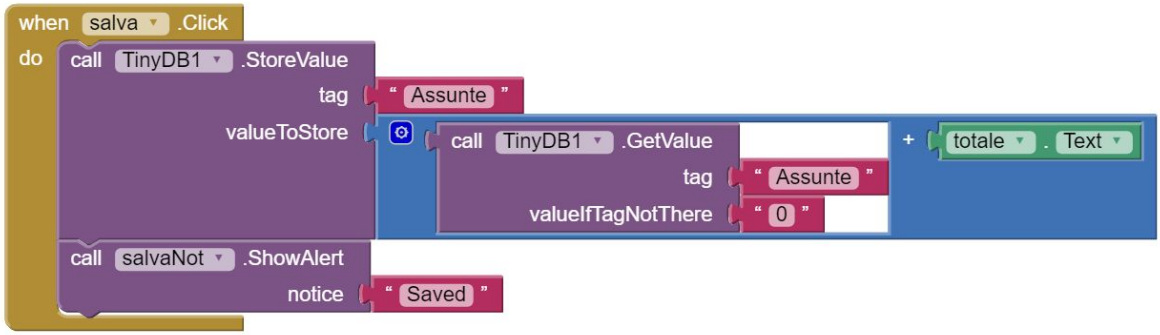
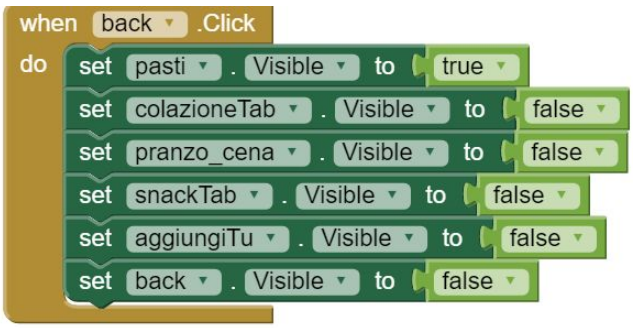
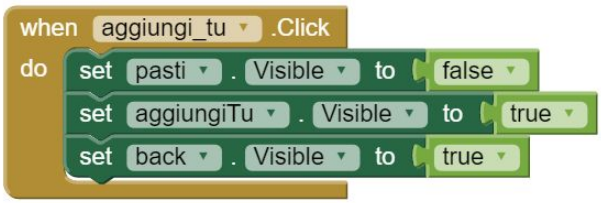
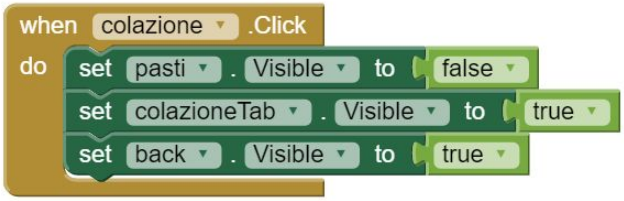


Figure 27

We put a checkbox "When ... click Do" and select the text in the empty space. After the DO we will connect the blocks shown in the picture. This block will be "set ... visible to" for both, followed by Logic False and True block.
This code sequence is used to hide or display components on the screen.







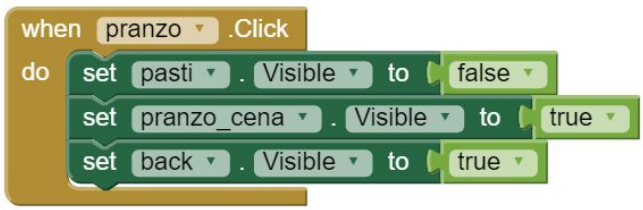
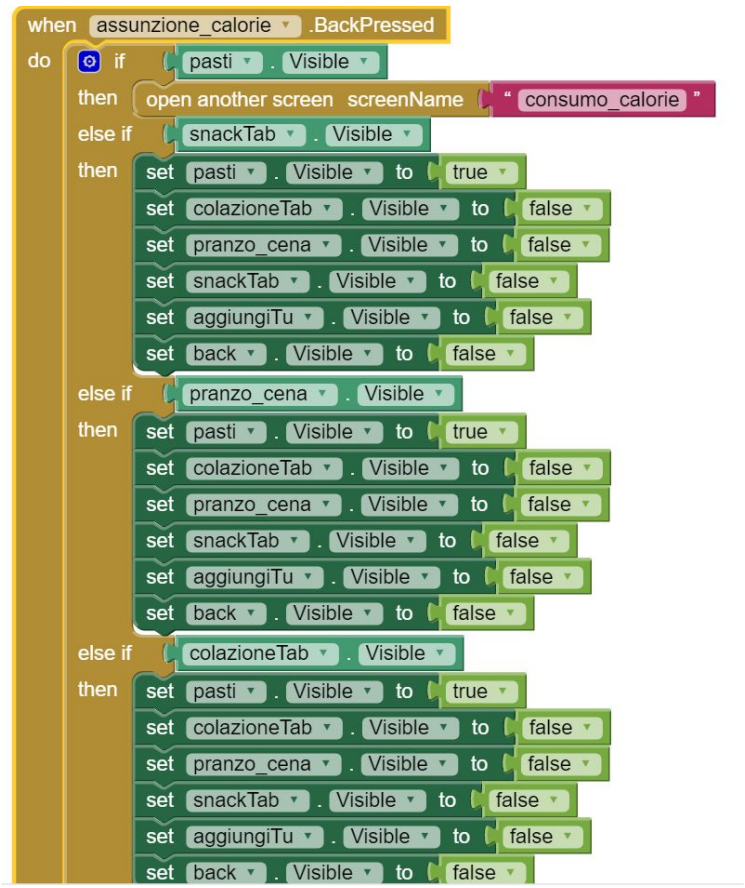


Figure 28

We put a checkbox "When ... click Do" and select the text in the empty space. After the DO we will connect the blocks shown in the picture. This block will be "set ... visible to" for both, followed by Logic False and True block.
This code sequence is used to hide or display components on the screen.



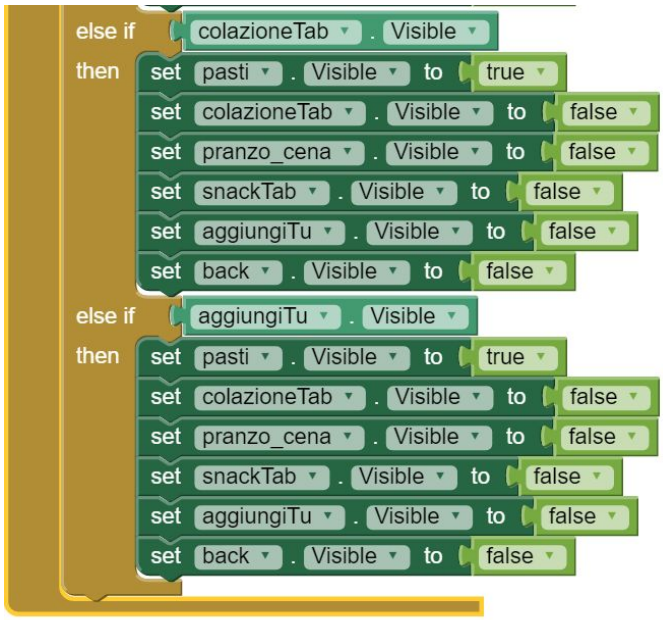


Figure 29

Lastly, we need to include a code block that allows us to change the colors in our application.

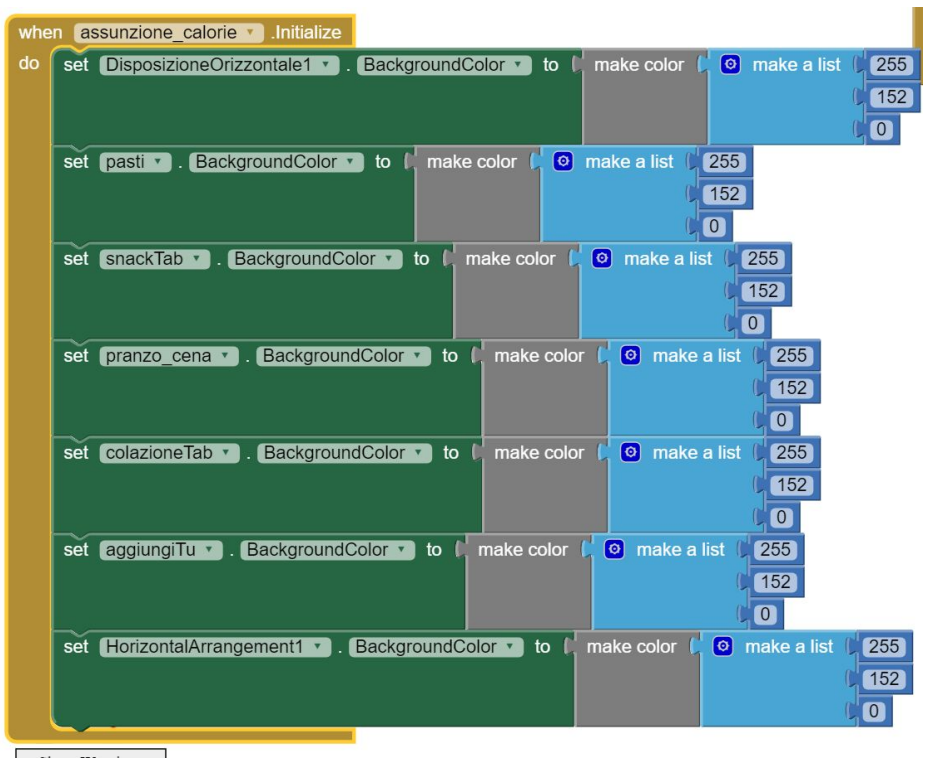


Figure 30

5. PEDOMETRO (pedoA)

This part of the application allows us to calculate the number of steps that have been taken during the day. The application to calculate the steps must remain open.

First, we will create the box containing the application logo that will allow us, once clicked, to return to the home of our application. We then insert a HorizontalArrangement in which we place a button with the logo image and a simple image with the pedometer logo (pedometer.png).
Next we will insert a HorizontalArrangement followed by a VerticalArrangement with two buttons named StartStop and N_steps respectively and a label called Foot_steps.
Next we insert another VerticalArrangement with two buttons, respectively reset and n_meters and a renamed label Meters.

Figure 31

Finally we put another VerticalArrangement. Within this we put a HorizontaArrangement with inside an image (etichettapedometro.png).
Next we insert another HorizontalArrangement with two labels named Save_meters and Save_nameters. Finally in this arrangement we put a sensor, specifically the pedometer.
Finally we include a notifier, pedoA_Notifier1, and a database, pedoA_TinyDB1.

BLOCKS:

This first block allows us to return to the Home when the application logo is clicked.

Figure 32

This block allows you to show the number of steps and distance traveled to the video.

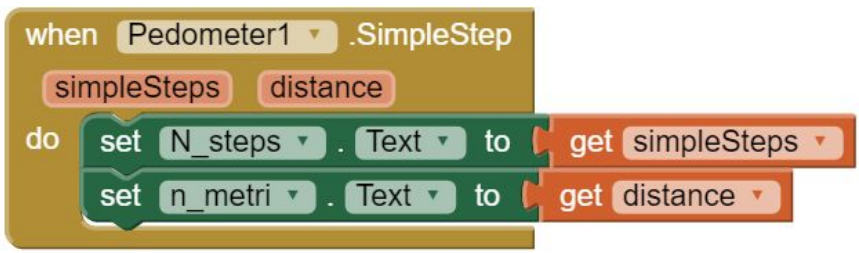


Figure 33

Let's now introduce a variable called PedometerAvviato that we initialize to the false value.

Figure 34

Given the above created variable, we create a block that allows us to start or stop the pedometer depending on the value of the variable itself.

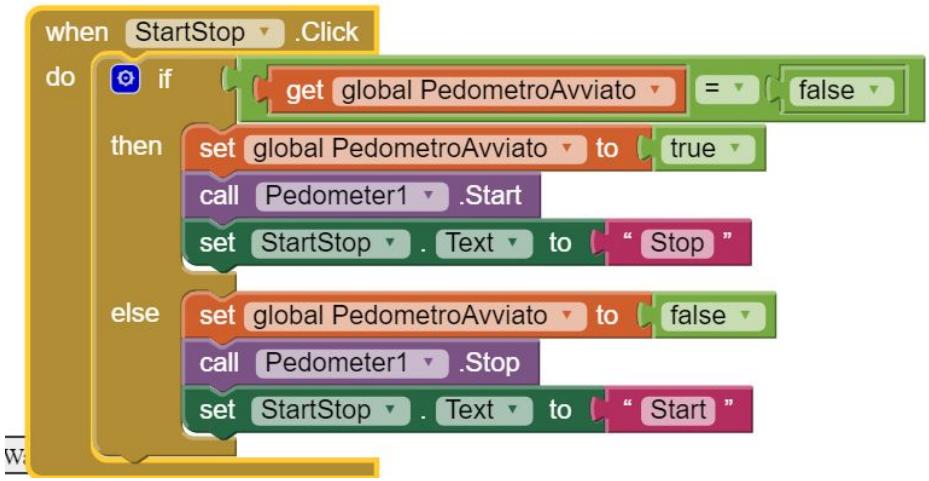


Figure 35

The next block allows us to show a notification with the ability to choose whether or not to save the data.

Figure 36

The following block allows you to save the passes and kms that are in the database according to the choice made in the previous notification.

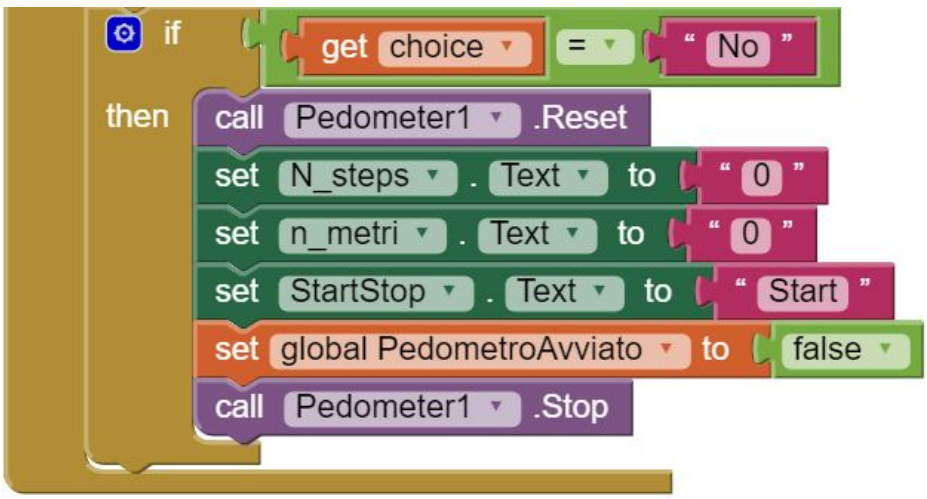
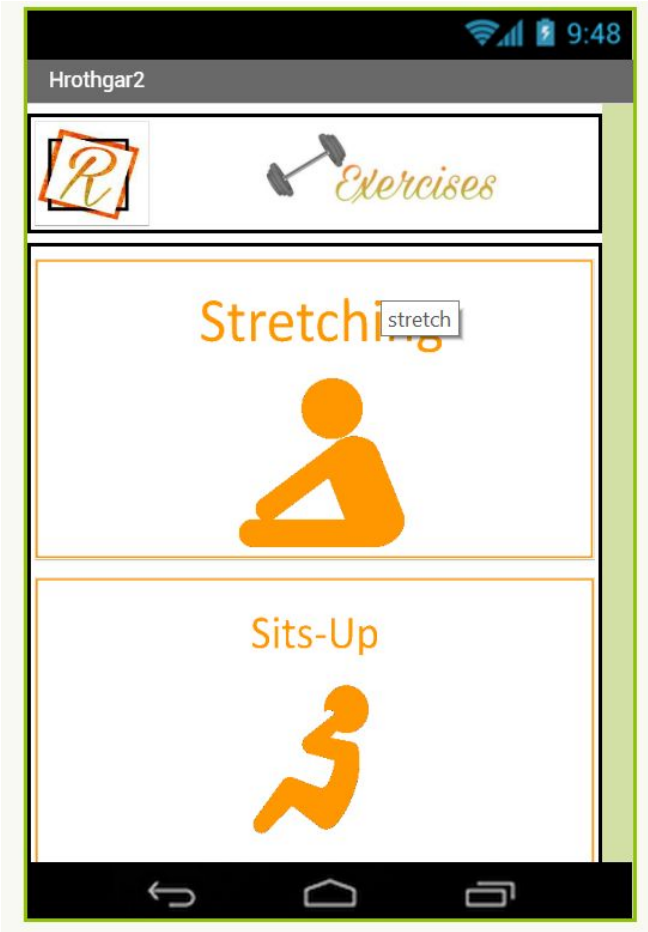


Figure 37

6. EXERCISES (screen2)

This part of the application suggests which exercises to carry out to consume excess calories.



First we put a HorizontalArrangement with a button that represents the application logo and an image (logo exercise2).
Next we insert a VerticalScrollArrangement containing five buttons, each of which will be called:
Stretch (stretching.png)
Abdominal (abdominals.png)
Squot (squat.png)
Push_ups (pushup.png)
Pull_ups (pullup.png)
Each of these buttons will lead to one page, so we will insert five VerticalScrollArrangement.

Figure 38

In the first, we will insert two labels, called pullupstyles and pullsticks (Anyone can benefit from learning how to pull-up. If you find that you need to build more strength, there are several exercises, as push-ups, you can practice to get strong Enough to start doing pull-ups. We recommend starting with 5 reps for each series.), And an image (pullups.jpg).
In the second arrangement, we insert two labels, denoted push_ups_txt and testo_push_ups (A basic push-up is an effective way to strengthen the chest and arm muscles. Simple push ups require no equipment and can be done anywhere. Reps of 4/5 series), and an image (pusa.jpg).
In the third we always insert two labels, label1 and squats2 (Squats are a full-body fitness staple that works on hips, glutes, quads, and hamstrings, and sneakily strengthen the core. Squats can help improve balance and coordination, as well as

bone density .), And an image. (Squat-11.jpg). In the fourth arrangement, we put two labels, stretching and textstretching (Stretching improves flexibility and makes up an important part of an exercise routine, and you have to be sure to do the correct stretches to prepare your body to do a particular activity. The web because if you do not do it right it may be self-defeating.), And an image (stretching.jpg). Finally, in the last arrangement, we include two labels, abdominal training and text_addominals (If you want to have sculpted abs the best is certainly doing sit ups. It is advisable to start with 10/20 reps for 2/3 series, but everyone must know his Limit and try improving day by day.) And an image (exercises_addominals_bass.jpg)

BLOCKS

The code of this app part is very simple.

This first block allows us to return to the Home when the application logo is clicked.

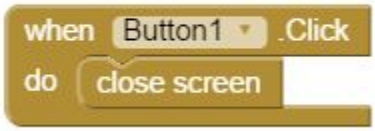
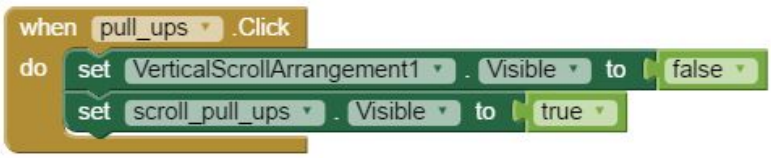
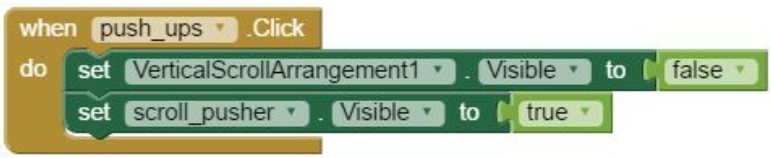


Figure 39

Successivamente andremo a inserire i blocchi che ci servono per far apparire o sparire determinati screen in base alle operazioni svolte dall'applicazione.





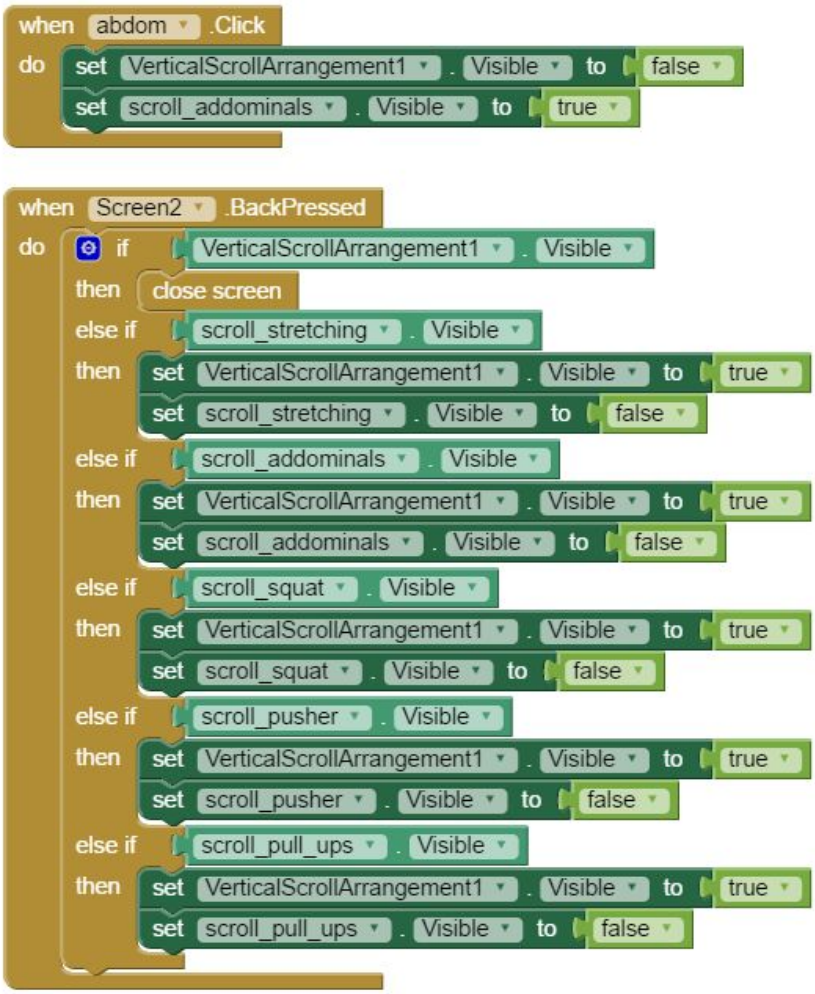

when abdom .Click
do set VerticalScrollArrangement1 . Visible to false
set scroll_addominals . Visible to true
when Screen2 .BackPressed
do if VerticalScrollArrangement1 . Visible
then close screen
else if scroll_stretching . Visible
then set VerticalScrollArrangement1 . Visible to true
set scroll_stretching . Visible to false
else if scroll_addominals . Visible
then set VerticalScrollArrangement1 . Visible to true
set scroll_addominals . Visible to false
else if scroll_squat . Visible
then set VerticalScrollArrangement1 . Visible to true
set scroll_squat . Visible to false
else if scroll_pusher . Visible
then set VerticalScrollArrangement1 . Visible to true
set scroll_pusher . Visible to false
else if scroll_pull_ups . Visible
then set VerticalScrollArrangement1 . Visible to true
set scroll_pull_ups . Visible to false

Figure 40

7. RECIPES (screen flags)

The last Screen we have to create is that of recipes in some countries. Each flag will open a new screen containing recipes for the selected country.

- We first put a HorizontalArrangement with a button that depicts the application logo and an image (recipes.jpg).
- We have six VerticalArrangement, one for each flag. We then rename them with the names of the countries corresponding to each flag:
  - Italy
  - Spain
  - Turkey
  - Greece
  - Portugal
  - Hungary

Figure 41

In Italy we put two pictures (pizza.jpg and carbonara.jpg) and four labels, pizza_testo and pizza_titolo and carbonara_testo and carbonara_titolo. Within the text labels we insert the pizza recipe (Pizza is a flat-breaded cheese usually topped with tomato sauce and cheese The modern pizza was invented in Naples (Italy). [1] The modern pizza was invented in Naples, Italy. , Italy, and the dish and its variants have since become popular and common in many areas of the world. With a normal portion, you assume 266 KCal.) And carbonara (Pasta alla carbonara is another dish of Italian tradition. Popular ingredients and strong flavors, and the type of pasta that is one of the most used is spaghetti but even linguine or other kind of short pasta are okay to be cooked. With a normal portion of 100 g, you assume 378 KCal.) .

In Spain we put two pictures (paella.jpg and tortilla.jpg) and four labels each label1, label2, label3 and label4. The label1 the paella call and the label2 will insert the paella recipe (Paella is a Valencian rice dish with ancient roots.) Valencian paella is

believed to be the original recipe and consists of white rice, bajoqueta and tavella, meat (chicken And rabbit), white beans, snails and seasoning such as saffron and rosemary. With a normal portion of paella, you assume 107 KCal.).

We rename the label3 tortilla espanola and in the label4 we will insert his recipe(Spanish omelette is the English name for a traditional dish from Spanish cuisine called tortilla espanola or tortilla de patatas. It is an omelette made with eggs and potatoes, sometimes also with onion and/or chives or garlic; fried in oil and often served cold as an appetizer. It is part of the cuisine of Spain and many South American countries, especially Argentina. With a normal portion of Spanish Omelette, you assume 105 KCal.).
In turkey we enter two images (shishkebab.jpg and donerkebab.jpg) and four labels: label5, label6, label7 and label8. The label5 we call it shis kebab and the label6 will contain its recipe (Shish kebab is a dish of skewered and grilled cubes of meat, it is popular in the whole of Asia, it is similar to a dish called shashlik, which is found in the Caucasus Region.It is usually made of lamb but there are also versions with beef or veal, swordfish and chicken meat. In Turkey, shish kebab and the vegetables served with it are grilled separately, normally not on the same skewer. The label7 will be called donerkebab and label8 will contain its recipe (Döner kebab is a type of Turkish kebab, made of meat cooked on a vertical rotisserie. Seasoned meat stacked in the shape of an inverted cone is slowly turned on the rotisserie, next to A vertical cooking element. The outer layer is sliced vertically into thin shavings as it cooks.)

Let us now go to Greece, which will also have two images (moussaka.jpg and friedcheese.jpg) and four labels: label9, label10, label11, label12. The label9 will be called moussaka and the label10 will contain its recipe (Moussaka is a eggplant or potato-based dish, often including ground meat, in the Balkans and Middle East, with many local and regional variations. Typically served hot. With a normal portion of Moussaka, you assume 103 KCal.). The label11 will be called fried cheese and label12 will contain its recipe (Fried cheese is a dish prepared using cheese that is fried in oil. Fried cheese can be dipped in a batter before frying and can be pan-fried or deep fried. Can be served as an appetizer or a snack.Fried cheese is typically served hot, right after being cooked, it may be accompanied by a dipping sauce or coated with a dressing.

In Portugal we find two images (baccalla.jpg and zuppaverde.jpge four labels: label2, label3, label4, label5. Label2 we rename Bacalhau and in label3 we insert the recipe (Bacalhau is the Portuguese word for cod and, in a culinary Fresh cod is referred to as fresh bacalhau (cod cod), cod, which has been dried without the addition of salt, is stockfish, and salt cod was a major export of the North Atlantic region, and It has become an ingredient of many cuisines around the Atlantic and in the

Mediterranean.With a portion of 100 g, you assume 103 KCal.) Label4 rename it hot green and in the label5 we put in its recipe (Caldo Verde is a popular Soup in Portuguese cuisine The basic traditional ingredients for hot green are potatoes, kale (although collard greens may be substituted), olive oil and salt. Onal favorite that has spread across the nation and abroad. It is surely a healthy soup.

Finally we go to Hungary where we put two pictures (zuppaungh.jpg and ciboungh.jpg) and four labels: label6, label7, label8 and label9. Label6 renaming it gulash and labeling7 we put its recipe (Gulash is a soup or stew of meat and vegetables, seasoned with paprika and other spices. Originally from the medieval Kingdom of Hungary, goulash is also a popular meal in Central Europe , Eastern Europe, the Netherlands, Belgium, Switzerland, Scandinavia and Southern Europe. There are 1134 KCal in a traditional cup of Gulash.). The label8 renames Halaszle and the label9 puts on its recipe (halaszle is a hot, spicy paprika-based river fish soup, originating from a dish of Hungarian cuisine, a bright red hot soup prepared with generous amounts of hot paprika and Carp or mixed river fish.With its generous use of hot paprika, halaszle is arguably one of the hottest (spicy hot) dishes native to the European continent. With one soup of 100 ml, you assume 44 KCal).

Subsequently, we will insert a new VerticalScrollArrangement and put inside it a TableArrangement. In the latter one we have six buttons each of which will refer to a flag:

- Pulsante_portugal
- Pulsante_grecia
- PulsanteTurkey
- Pulsante_italy
- Pulsante_spain
- Pulsanteungheria

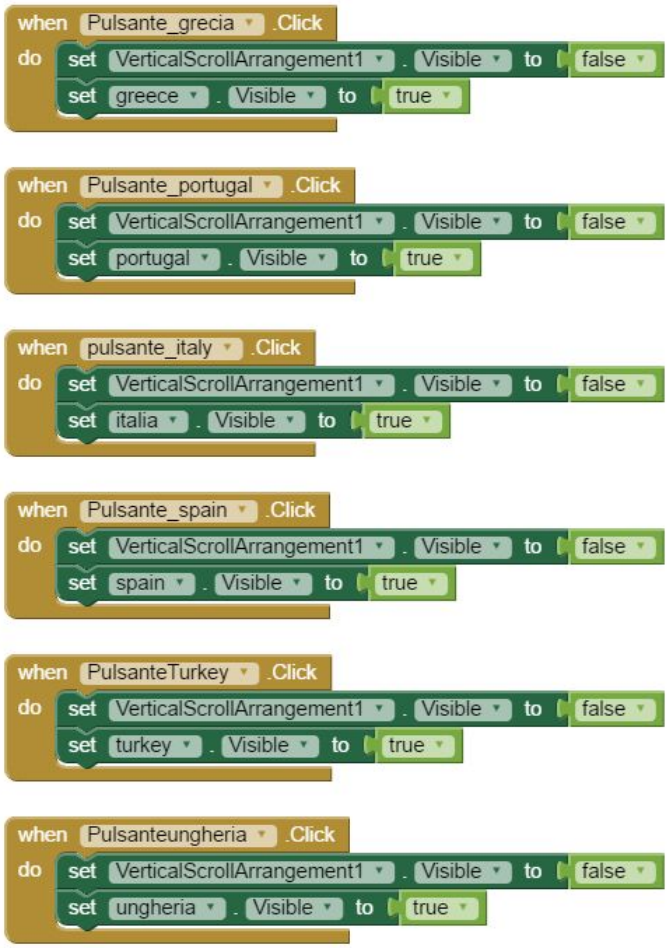
BLOCKS

The code of this app part is very simple.

This first block allows us to return to the Home when the application logo is clicked.

Figure 42

Next, we will enter the blocks we need to make certain screens appear or disappear based on the operations performed by the application.




when flags8 .BackPressed
do if VerticalScrollArrangement1 . Visible
then close screen
else if italia . Visible
then set VerticalScrollArrangement1 . Visible to true
set italia . Visible to false
else if spain . Visible
then set VerticalScrollArrangement1 . Visible to true
set spain . Visible to false
else if turkey . Visible
then set VerticalScrollArrangement1 . Visible to true
set turkey . Visible to false
else if greece . Visible
then set VerticalScrollArrangement1 . Visible to true
set greece . Visible to false
else if portugal . Visible
then set VerticalScrollArrangement1 . Visible to true
set portugal . Visible to false
else if ungheria . Visible
then set VerticalScrollArrangement1 . Visible to true
set ungheria . Visible to false

Figure 43