schwyz university of teacher education

TurtleStitch Workshop – Introductory Workbook for Students and Educators

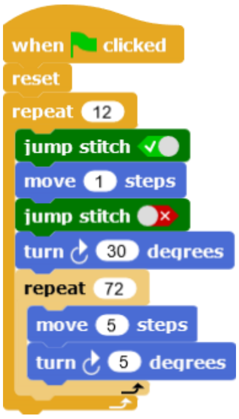
1 Introduction

This introductory manual shall provide students and educators an easy start to TurtleStitch.

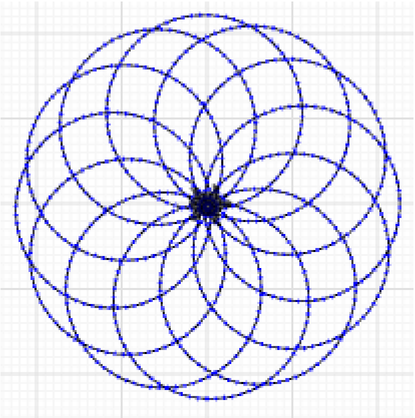
TURTLE STITCH

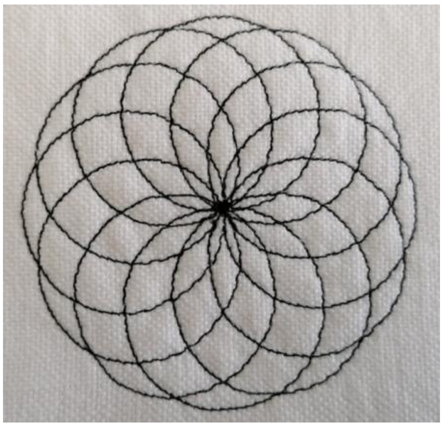
TurtleStitch is a block-based programming language that can be used from your internet browser. Therefore, you do not need to learn complicated code but instead you can jump in directly into the fun part of programming!

Using TurtleStitch, you can design a multitude of forms and figures. For example, this code (left screenshot) creates the respective pattern on the screen (middle screenshot), which can finally be embroidered on fabric using an embroidery machine (right screenshot).



```
when ⚑ clicked
reset
repeat 12
  jump stitch ✓
  move 1 steps
  jump stitch ✗
  turn ↻ 30 degrees
  repeat 72
    move 5 steps
    turn ↻ 5 degrees
```





Before you start stitching, be sure to understand the settings and everything about your embroidery machine. If you want to learn TurtleStitch now and have a look at the embroidery machine later, you can go on with this manual and we wish you lots of fun doing that 😊

Schwyz University of Teacher Education
Beat Horat, Junior Researcher
Zaystrasse 42 CH-6410 Goldau
T +41 41 859 05 90
beat.horat@phsz.ch
www.phsz.ch

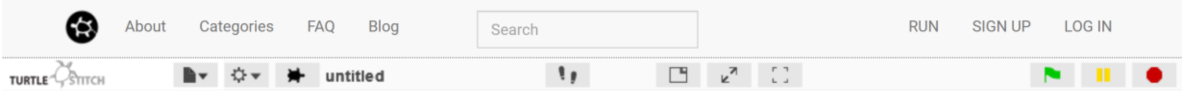
2 Starting with TurtleStitch

2.1 Start the Program

Navigate to the website https://www.turtlestitch.org/run. TurtleStitch can be used directly in your browser, you can save your embroidery patterns and many more without installing a software.

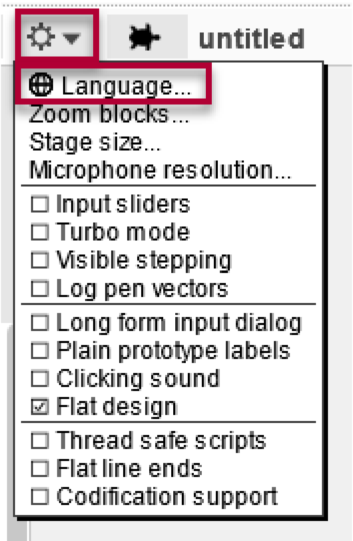
2.2 Language

In the upper part of TurtleStitch, you see a wide frame with a variety of buttons and settings.



We will return to some details later, as you certainly want to start programming, we discuss only the most important settings now.

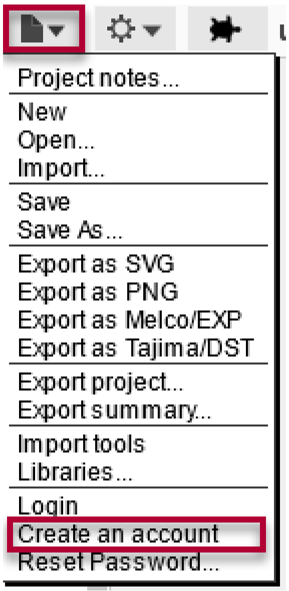
First, choose the language that you want to use in TurtleStitch. Klick on the cogwheel symbol and choose “Language”.



From the list that opens, choose your preferred language such as “English”. The display language of TurtleStitch will immediately switch to what you just chose. The language selection will remain if you use the same browser.

2.3 Open an Account

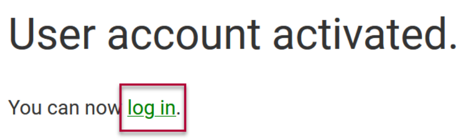
To save your work and later use it again, it is best to create a user account. Klick on the document symbol and choose “Create an account”. Then choose a username and state your email address and password.



You need to accept «Terms of Service» and «Privacy Policy» to create an account. Moreover, you need to subscribe to the newsletter, which will be sent very rarely.

You will receive a confirmation email and you need to confirm your account creation by clicking on the link in that email.

Your user account is now ready to use. Click on “log in” and provide your login credentials you just determined.

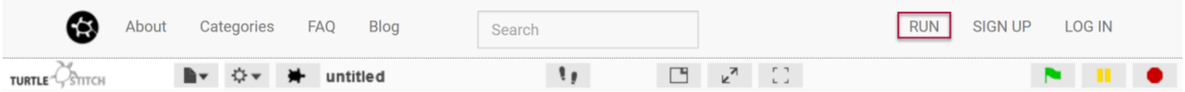


User account activated.

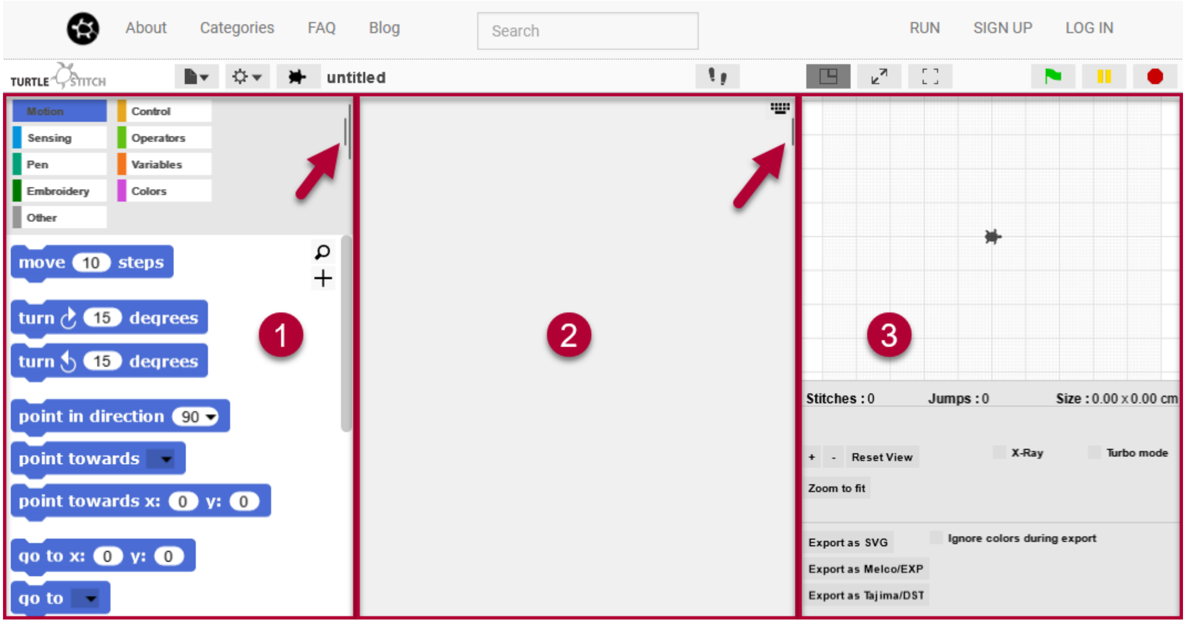
You can now log in.

2.4 Overview

Whenever you click on “RUN”, you get back to the main screen where you program your code.



The main screen is split in three parts:

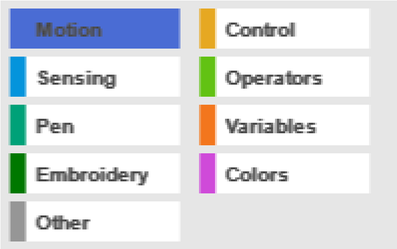


1) On the left-hand side, you see the palette with code blocks. You can choose your code blocks from here and click on them to test the code blocks.
2) In the middle of the screen, you have your scripting area, also called coding area. For coding, simply drag and drop the necessary blocks from the palette to the coding area.
3) On the right-hand side, you see the stage featuring the mascot of TurtleStitch, which is a turtle. This turtle will do exactly what you tell it to do in your coding area. Below the stage, you see some additional buttons and functions. For example, you can export your code to an USB stick and many more. We will have a look at this later.

You can change the size of the three parts when you drag and drop the symbol with the two vertical stripes, indicated with red arrows in the screenshot.

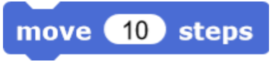
2.5 Code blocks

We can choose our code blocks from the palette. They are split into different categories that we should know about. The categories are colored to make overview easier.



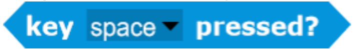
The names of the categories are self-explanatory and the blocks within are best learned when used. Therefore, we provide a broad overview with examples here:

- Motion: When you choose this category and for example click on the block “move 10 steps”, your turtle is already moving. Click on some of the blocks to test them.

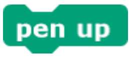
  ```
  move 10 steps
  ```

- Sensing: Your turtle can sense things by touching them or asking them questions. For example, when using this block, the turtle asks if the space bar is pressed.

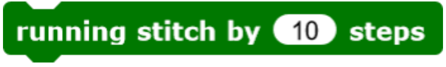
  ```
  key space pressed?
  ```

- Pen: These blocks are used to paint and draw. In other words, the blocks decide whether the turtle draws something when moving and what it draws. For example, “pen up” would mean that nothing is drawn when the turtle moves.

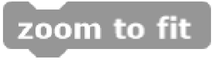
  ```
  pen up
  ```

- Embroidery: As we want to stitch our patterns later, this is an important category for us. When we for instance choose a running stitch, all subsequent movements will be drawn in this embroidery pattern. This also means that choosing your embroidery pattern does not do anything right away but will be used upon the next movement.

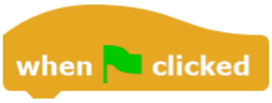
  ```
  running stitch by 10 steps
  ```

- Other: These are special commands. Now there is only “zoom to fit”, which adjusts your visible stage to the area that contains stitching.

  ```
  zoom to fit
  ```

- Control: This is a selection of very exciting blocks, which we will use often. First, you will need “when ‘green flag’ clicked”. This is the starting command in TurtleStitch for programs to run and start stitching. Other blocks will make decisions or repeat commands and we will look at them soon.

  ```
  when clicked
  ```

- Operators: The operators can do calculations for you. You can make them add numbers and many more.

- Variables: These blocks make your life much easier when coding. For example, create a variable “Counter” that checks how many times a certain event happens.

- Colors: You can set the color of the thread on the screen that shall be used.

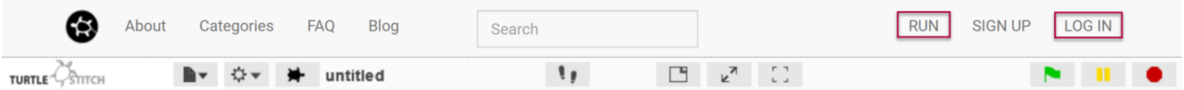
3 My First TurtleStitch Program

Now we are ready to code our first TurtleStitch Program together 😊

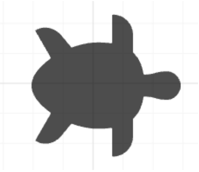
3.1 Prepare Your Main Screen

Log in to your account before you start. This is important to save your programs.

Klick on “LOG IN” and provide your account credentials in case you are not already logged in. Then, klick on “RUN” to return to the main screen.



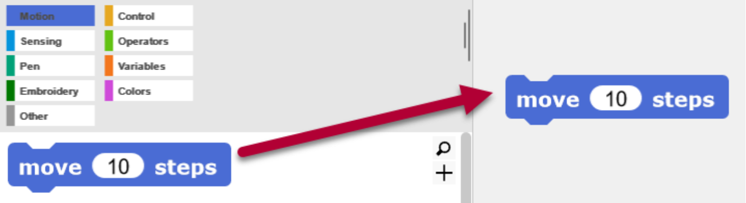
The turtle waits on the stage and is pointing to the right. The coding area is empty.



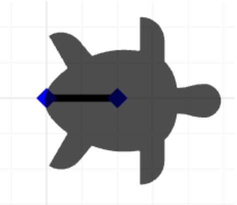
Now we are ready!

3.2 Motion

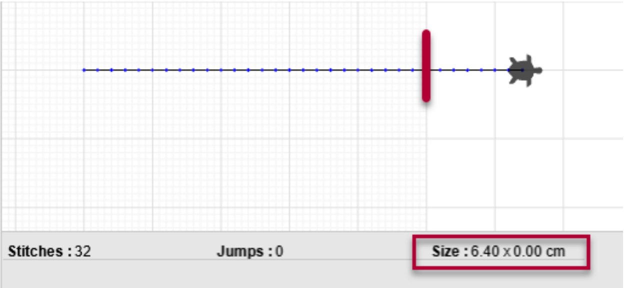
From the palette on the left-hand side, choose the block “move 10 steps” from the “Motion” category. Drag and drop the block to the coding area.



One step is 0.2 mm, 10 steps are 2 mm accordingly. When you click the block, the turtle on the stage moves 2 mm in the direction that it is pointing to now. **Please remember that the turtle does everything from its own point of view.** This means that when you tell the turtle to move forward, it will move to the right on the screen when it is pointing to the right already. For the turtle, this is where “forward” is in this example 😊



When you click on the block several times, the turtle moves 2 mm every time. It makes a stitch every 2 mm while moving. At some point, the turtle will leave the stage but goes on stitching anyway. You can see the edge of the stage when you look closely, the stage is colored in light gray with a dense square pattern and the outer part is white.



Stitches : 32 Jumps : 0 Size : 6.40 x 0.00 cm

You can see important information below the stage:

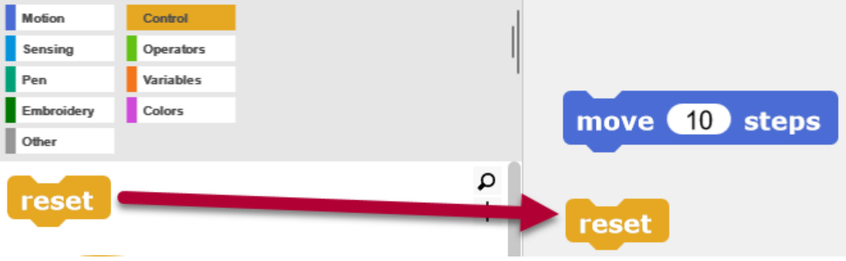
- Stitches: Counts the stitches already made. The stitch count hints to how much time the later embroidery will take to be created on the embroidery machine.
- Size: The size of the pattern that you stitched so far tells you how much space the embroidery will take up and if the stitching frame, fabric etc. is large enough for your embroidery. In general, stay inside the stage in TurtleStitch and your embroidery should fit conventional stitching frames.

3.3 Reset the Stage

When you develop patterns in TurtleStitch, you will have to start your code again and again. This means that you will remove the patterns you created from the stage every time before you try your new version of the code.

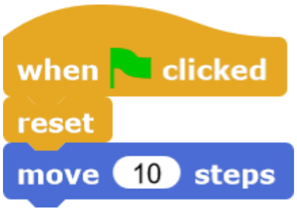
Drag and drop the “reset” block from the “Control” category to you coding area. When you click on it, the pattern on the stage disappears and the turtle moves back to the initial position in the middle of the stage and points to the right.

While the stage is reset, the coding area does not change when you use the “reset” block and you can continue coding and testing.



3.4 Put Everything Together

As mentioned, the starting point in most programs is “when ‘green flag’ clicked” and this is a useful signpost for starting your code. You find the block in the orange “Control” category. Snap it on top of your code like this:



```
when 🏁 clicked
reset
move 10 steps
```

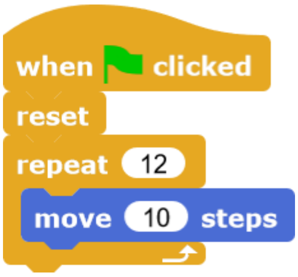
Click on the green flag and first your stage is reset and then your turtle moves 10 steps. Change the count of steps to 20 and see what happens when you click the green flag again. Try other numbers of steps as well.

The turtle always starts from the beginning and does whatever you order it in your code.

By the way, it is totally normal to execute your code again and again. This, way you can always test your code and if the latest change to the code does what you wanted. Even professional programmers do this, and it is totally normal to test your code countless times while programming.

3.5 Repeat Loop

You need to be careful that the stitch length is not too long, otherwise the thread will be very loose. For stitching a longer section but keep the short stitching length, repeat the “move 10 steps” block several times by nesting it into a “repeat 12” block. Change the iterations (repetitions) and try different step counts to see the difference in your result.

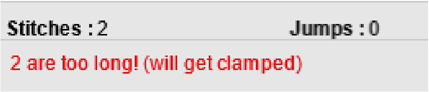


```
when 🏁 clicked
reset
repeat 12
  move 10 steps
```

3.6 Stitching Settings

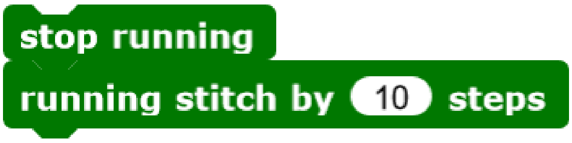
It is highly recommended that you choose your stitching settings right from the start in your code. This means you set stitch length and stitch type and only then you start moving your turtle. This way, you can prevent your turtle from making too long stitches when moving great distances.

Beneath the stage, you would see a warning in case you created too long stitches that would get clamped:



```
Stitches : 2                    Jumps : 0
2 are too long! (will get clamped)
```

We use a running stitch in our example that creates a stitch every 10 steps (2 mm), even if the turtle moves a hundred steps at once. Please note that 10 steps are the recommended stitch length with which you should start until you have more experience with embroidery.



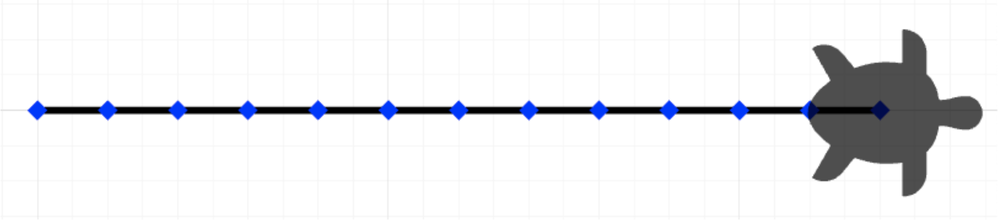
```
stop running
running stitch by (10) steps
```

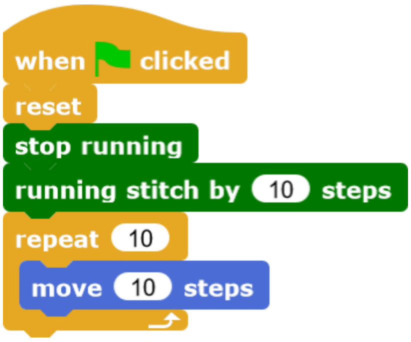
Please visit chapter 4.1 “Stitching Types” for more descriptions on the matter.

The block “stop running” may be used at the start of your program for removing all settings that come from the previous runs of your code and that could mess with your settings you need at the start of your code.

3.7 Stitch a Line

We now repeat and put together what we learned so far. To do this, we will stitch a line of 24 mm length. Follow the steps below, create the code blocks and snap them together according to the example.





```
when 🏁 clicked
reset
stop running
running stitch by (10) steps
repeat (10)
  move (10) steps
```

The “reset” block initializes all setting, the turtle moves to the initial position (0, 0) and points to the right, the stage is cleared. Using “stop running” resets the stitching settings that might have been used before from the “Embroidery” category.

Using “running stitch by _ steps” sets the stitch type and stitch length. A running stitch is a simple straightforward stitch with 0.2 mm length per stitch.

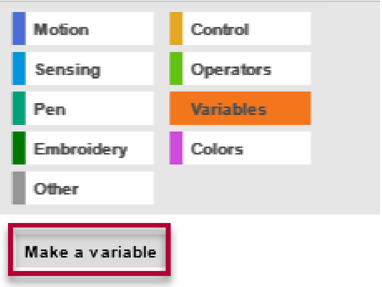
The block “repeat _” executes the blocks that it embraces as many times as stated in the repeat block.

“move _ steps” declares, how long the stitched line will be. Again, the step will be 0.2 mm. The start and the end of the step will consist of a stitch each. These are indicated on the stage by the blue diamond symbols.

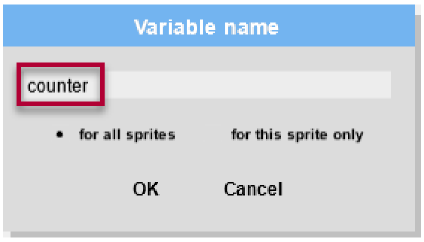
Create the code block and experiment with it for fun!

3.8 Variables

You can simplify the above example by using variables. Choose the “Variables” category in the palette and click on the grey “Make a variable” button:



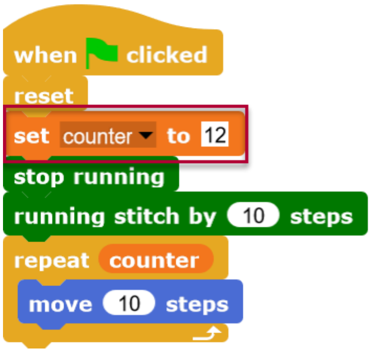
Choose a name for your variable and then click “OK”:



counter

We can now change our program. The drag and drop the variable “counter” from the palette into the “repeat 12” loop and replace the number 12. The loop now reads “repeat counter”.

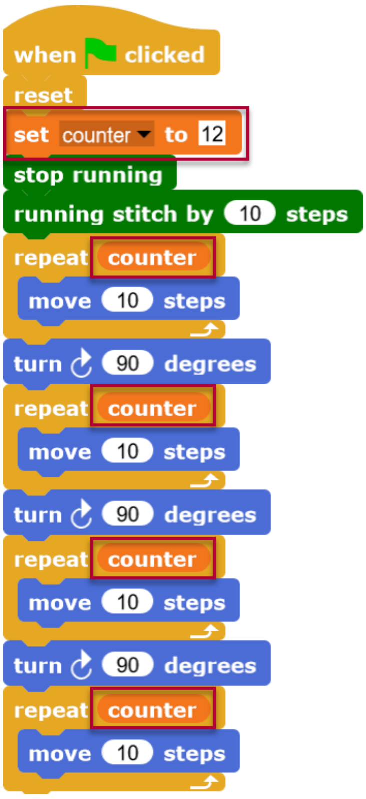
Now we need to set the counter as it has no value yet. Let us set it to 12 and the code will do what it did before. By the way, set a value for a variable is called initializing.



```
when clicked
reset
set counter to 12
stop running
running stitch by 10 steps
repeat counter
    move 10 steps
```

The advantage of variables is, that you can used them many times, but you need to set a value only once.

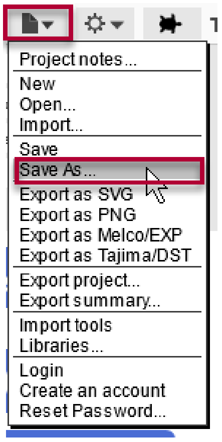
For example, let us create a square. We set the length of the edges only one time, but we use it for all four edges. When the square need to be bigger, increase the length of the edges one time and all four are changed at once:



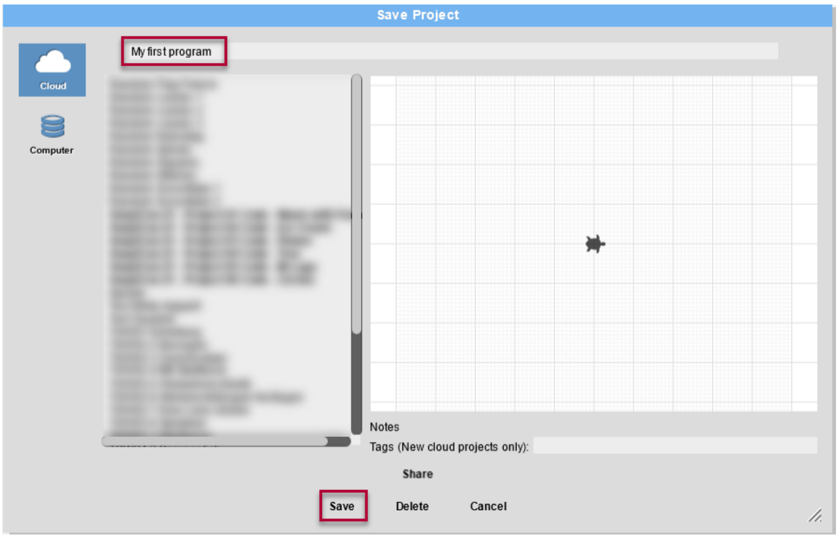
```
when ⚑ clicked
reset
set counter ▾ to 12
stop running
running stitch by 10 steps
repeat counter
    move 10 steps
turn ↻ 90 degrees
repeat counter
    move 10 steps
turn ↻ 90 degrees
repeat counter
    move 10 steps
turn ↻ 90 degrees
repeat counter
    move 10 steps
```

3.9 Save Your Work

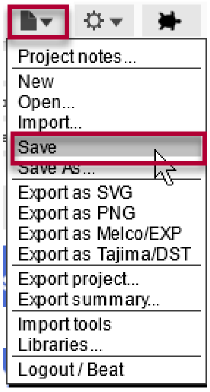
You need to be logged in to save your work. Klick on the document symbol and choose “Save As” from the menu:



Choose a name for your program and click “Save”:



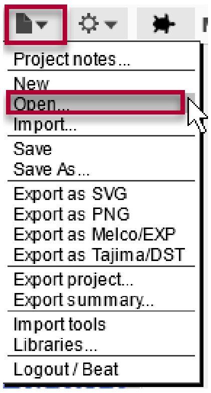
When you adjust your program that already has a name, you can simply choose the document symbol again and now choose “Save”:



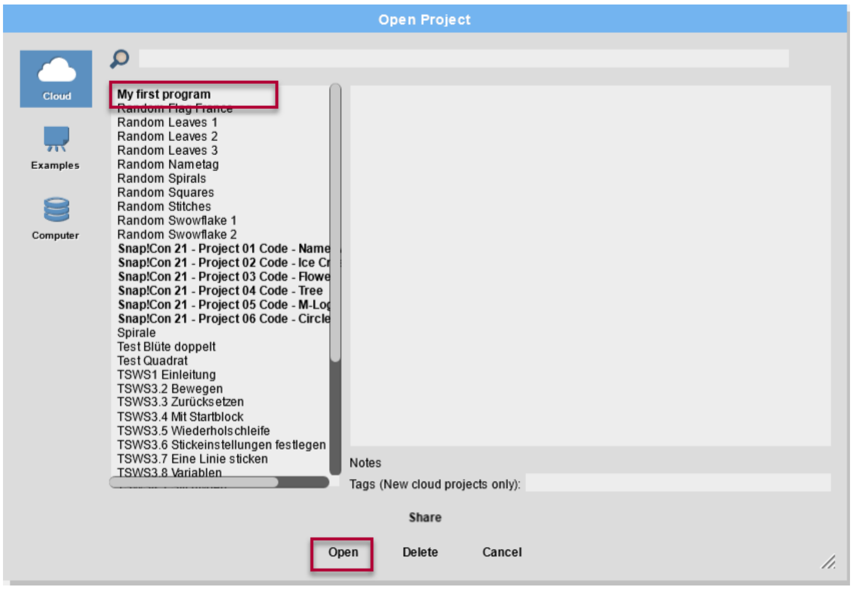
Attention: Saving means that you can open the program later in TurtleStitch. When you want to stitch your pattern on an embroidery machine, see chapter 7 “Using Embroidery MachineUsing Embroidery ” for instructions on how to do that.

3.10 Opening Projects

Click on the document symbol and choose “Open”:



Choose a program from the list and click “Open” below the list:



3.11 Share Projects

Your project will be shared with the community when you save it the first time. Therefore, your programs are visible to everybody in TurtleStitch that has an account.

Your shared projects will be displayed in bold letters in the list when choosing “Open”. In case you do not want to share a project, click on it, and choose “Unshare” from the bottom of the list. Sharing an unshared project is very similar, just click on it and choose “Share”.

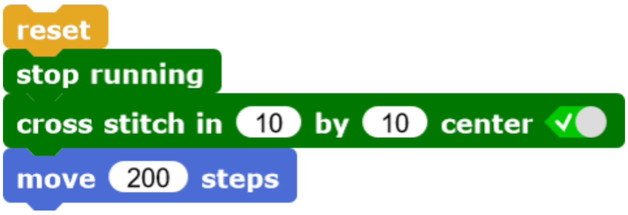
4 Further Hints and Simplifications

We provide some hints about stitching patterns here and how you can make your code simpler and clearer.

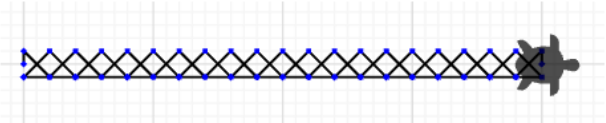
4.1 Stitching Types

As explained in chapter 3.6 “Stitching Settings”, choose your stitching settings before you move your turtle. Choosing the stitching setting does not create a visible result right away. Only when the turtle moves, it does stitch in the style you just chose.

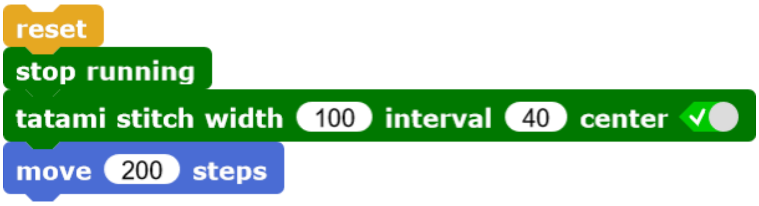
The stitches can vary greatly. See here how the cross stitch looks on the stage:



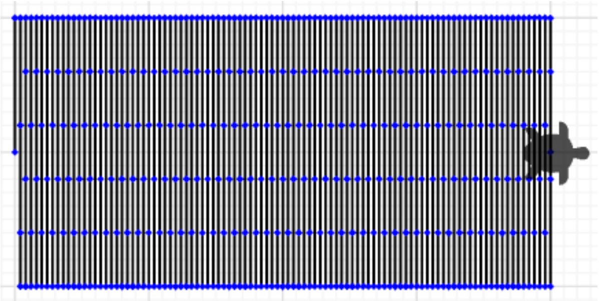
```
reset
stop running
cross stitch in (10) by (10) center ✓
move (200) steps
```



And now see an example for the tatami stitch:



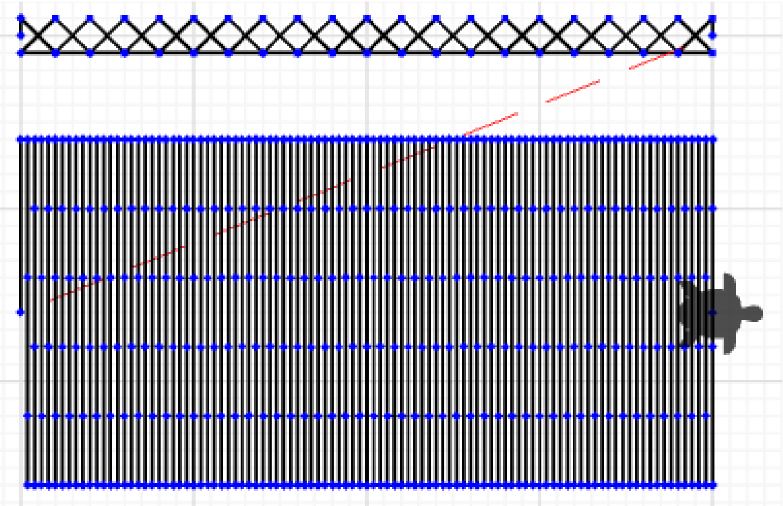
```
reset
stop running
tatami stitch width (100) interval (40) center ✓
move (200) steps
```



Looks great, doesn’t it? Test some other stitching types for yourself and see what you like, also play with the settings of the stitches to find out what they do.

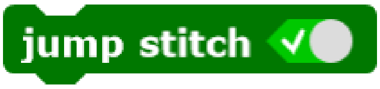
4.2 Jump Stitch

A special and important stitch in TurtleStitch is the jump stitch. Let us assume that you want to stitch the above example stitches beneath each other for comparison. It would look like this:



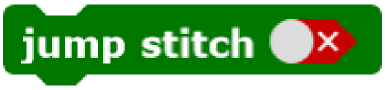
Normally, the turtle stitches when it moves, as seen in many previous examples. However, we find a red line in this example that indicates that the turtle did not stitch along that line. Our turtle jumps from one place to another, which is therefore called a jump stitch. The thread will be knotted before the jump and after, so the red part in between the jump and the landing can be cut out with scissors later.

Mark the place where your turtle must jump with a “jump stitch” block. The switch set to green can be loosely translated to “jump = yes”:



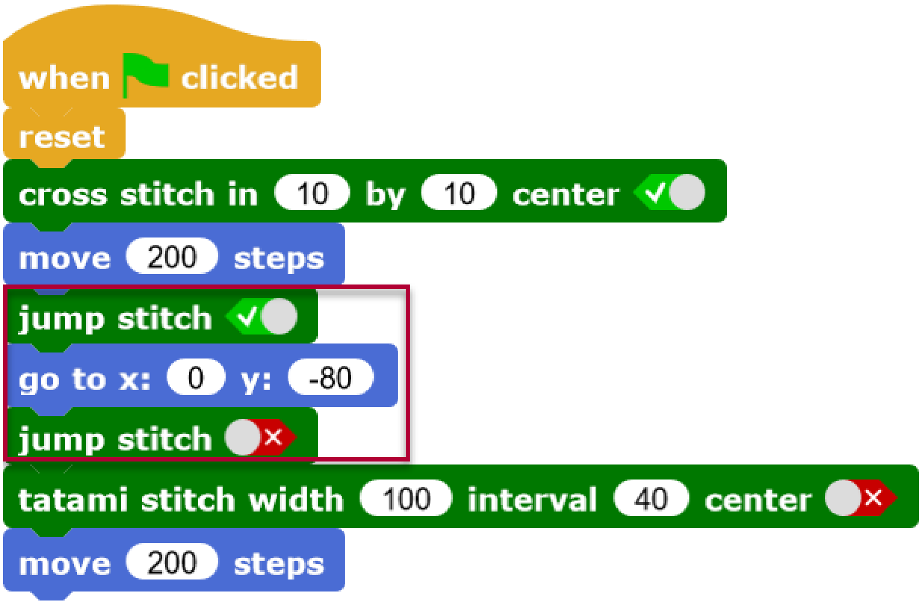
jump stitch ✓

After we moved the turtle to the landing site, we tell it to land using another “jump stitch” block. This time we set the switch to red, “jump = no” would then mean “land”:



jump stitch ✗

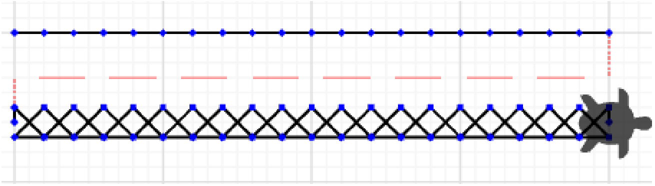
We need to match jumps and landings, so they always match. This is an example of a complete jump stitch with movement in between jump and landing:



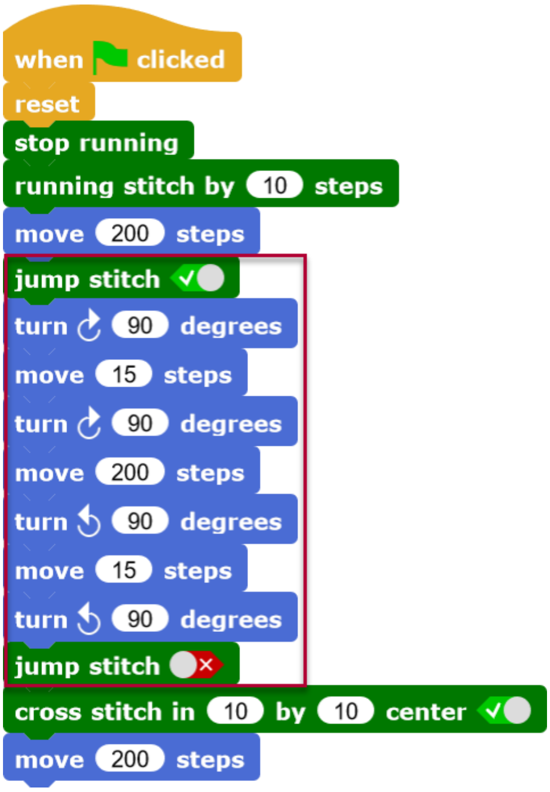
```
when 🏁 clicked
reset
cross stitch in (10) by (10) center ✓
move (200) steps
jump stitch ✓
go to x: (0) y: (-80)
jump stitch ✕
tatami stitch width (100) interval (40) center ✕
move (200) steps
```

4.3 Create Personal Coding Blocks

Imagine that you want to compare stitch types. We always jump to the start of the line for each stitch type and little bit further down, as if we would write a text:



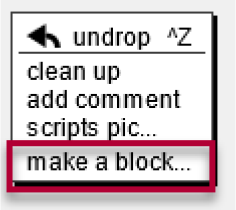
The code could look as in the following example:



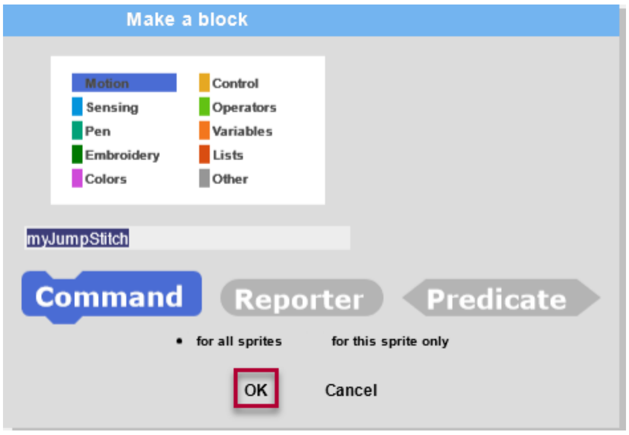
```
when 🏁 clicked
reset
stop running
running stitch by (10) steps
move (200) steps
jump stitch ✓
turn ↻ (90) degrees
move (15) steps
turn ↻ (90) degrees
move (200) steps
turn ↺ (90) degrees
move (15) steps
turn ↺ (90) degrees
jump stitch ✗
cross stitch in (10) by (10) center ✓
move (200) steps
```

The part marked with a red frame could be repeated now before we stich an additional line. This is tedious to rebuild, but there is a much easier way. We can create a block out of this sequence and reuse it any time we want.

Click with your right mouse button on the coding area and choose “make a block”:



Choose a category. Our block is mainly about motion so “Motion” is the fitting category here that we click on. Then, provide a name for your new block and click “OK”:

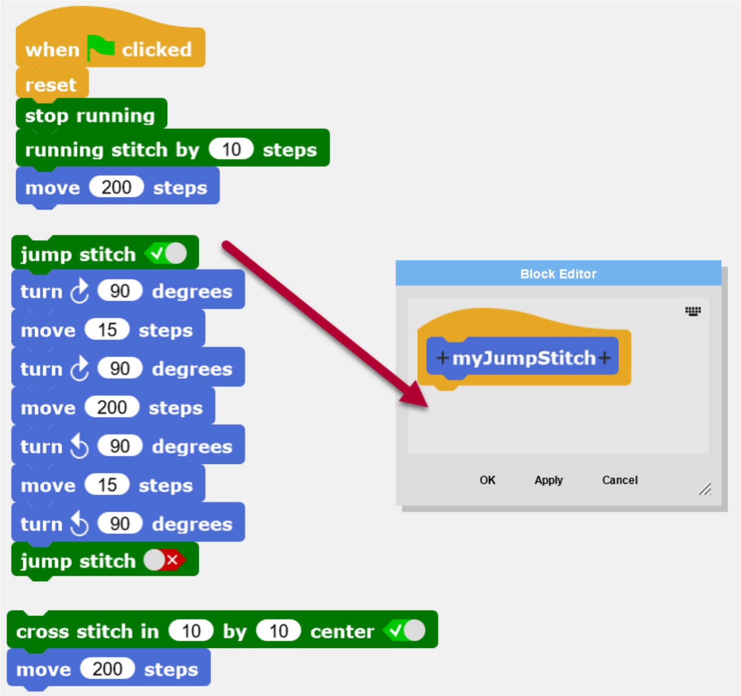


myJumpStitch

● for all sprites ○ for this sprite only

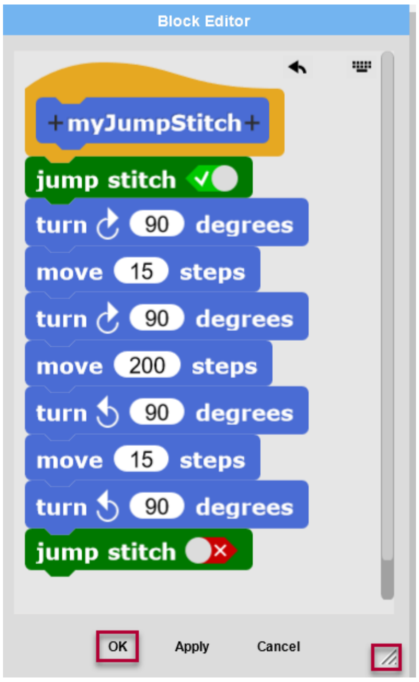
Now split the previous code so that you can move the jump part individually. Now drag and drop this part into your new block.

You could create code from the palette and build it up in your new block directly. Since we already got this part in our example, we reuse it of course. Reusing working code is a common programmers’ strategy and there is nothing wrong about it, so use this principle in your coding when you can:



```
when clicked
reset
stop running
running stitch by 10 steps
move 200 steps

jump stitch ✓
turn ↻ 90 degrees
move 15 steps
turn ↻ 90 degrees
move 200 steps
turn ↺ 90 degrees
move 15 steps
turn ↺ 90 degrees
jump stitch ✗

cross stitch in 10 by 10 center ✓
move 200 steps
```

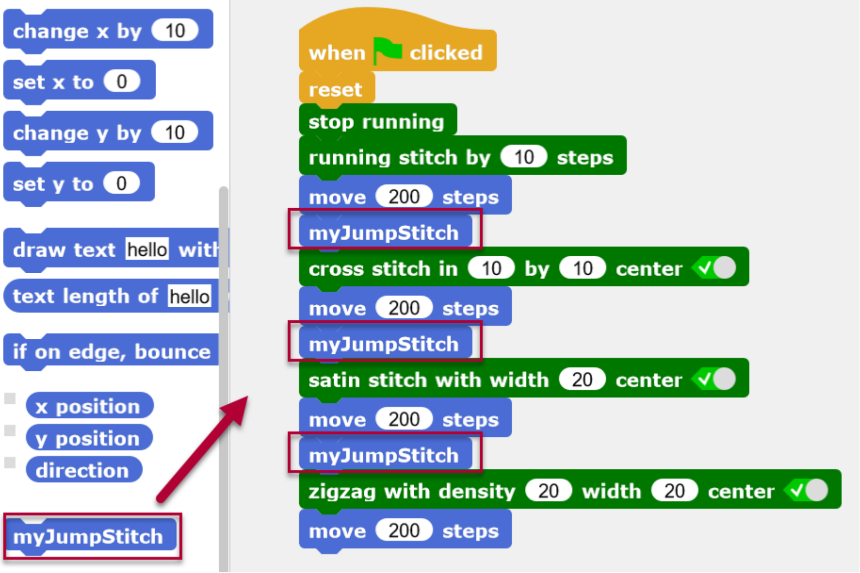
You can resize your block editor window by drag and drop the small triangle symbol in the bottom right corner. When you are happy with your block, click “OK” for saving the code:



```
+ myJumpStitch +
jump stitch ✓
turn ↻ 90 degrees
move 15 steps
turn ↻ 90 degrees
move 200 steps
turn ↺ 90 degrees
move 15 steps
turn ↺ 90 degrees
jump stitch ✗
```

After saving it, you can find your block in the category to which you assigned it. We created a blue “Motion” block, and you will thus find it in that category on the bottom the list.

We can now use the new block and replace the original sequence, which will be much shorter and easier to understand. Moreover, we can reuse the block in two further instances, which is of course immensely practical and timesaving:



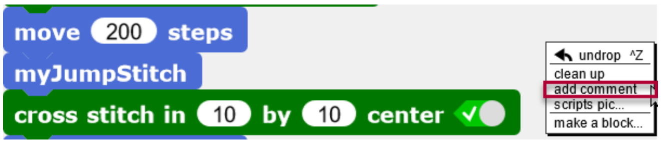
```
change x by 10
set x to 0
change y by 10
set y to 0
draw text hello wit
text length of hello
if on edge, bounce
x position
y position
direction
myJumpStitch
```

```
when 🏁 clicked
reset
stop running
running stitch by 10 steps
move 200 steps
myJumpStitch
cross stitch in 10 by 10 center ✓
move 200 steps
myJumpStitch
satin stitch with width 20 center ✓
move 200 steps
myJumpStitch
zigzag with density 20 width 20 center ✓
move 200 steps
```

4.4 Using Comments

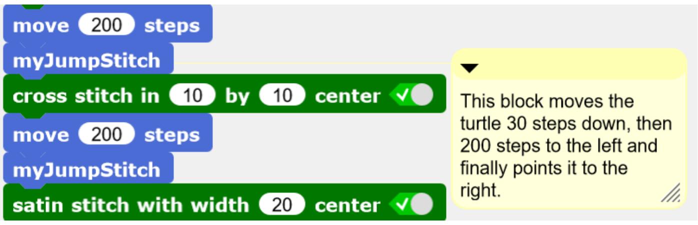
We can add a comment to the block that we just created. This will increase readability of our code significantly. Other stitchers will understand our program better, which is true for ourselves as well when we return to our code after some time passed.

Right click on the coding area and choose “add comment”:



```
move 200 steps
myJumpStitch
cross stitch in 10 by 10 center ✓
```

Type your description in the comment window. You can also drag and drop the comment windows to a specific block of your code to link both together, which is indicated by a fine yellow line. When you do this, it is crystal clear which part of the code you commented.



```
move 200 steps
myJumpStitch
cross stitch in 10 by 10 center ✓
move 200 steps
myJumpStitch
satin stitch with width 20 center ✓
```

This block moves the turtle 30 steps down, then 200 steps to the left and finally points it to the right.

5 Test and Debugging

Sometimes code does not quite do what the programmer wanted. This is completely normal and most of the times it is just minor mistakes and inaccuracies that caused the problem. The strategies to find these errors are called debugging, as we want to find the “bug” in our code 😊

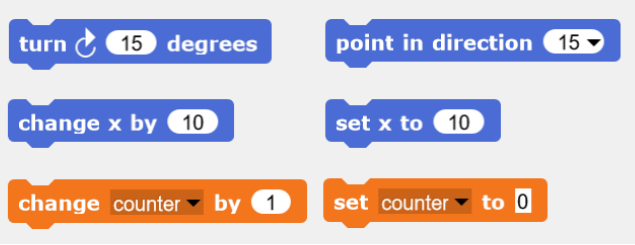
Use this checklist to find the most common errors quickly.

5.1 Forgotten Block

Check the blocks that you used individually. Do they make sense together? Is there something missing? In case you write down another example code for reusing it, check if you missed something by simply counting the blocks in the original compare it to your version. Of course, it helps when someone else is helping as four eyes see more than two.

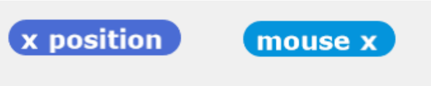
5.2 Mix Up Blocks

Maybe you mistakenly used the wrong block. Very often, similar blocks are mixed up, for example using relative movement instead of absolute positions and directions. Compare the blocks below, the left is slightly different the one on the right:



```
turn ↻ 15 degrees          point in direction 15
change x by 10             set x to 10
change counter by 1        set counter to 0
```

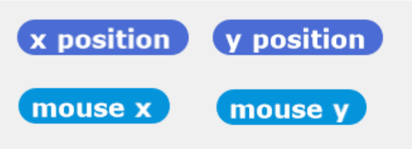
Also, the colors of the blocks give away what their function is. These examples are different, what you can already see from the color:



```
x position      mouse x
```

5.3 Mixing Up X-Coordinates and Y-Coordinates

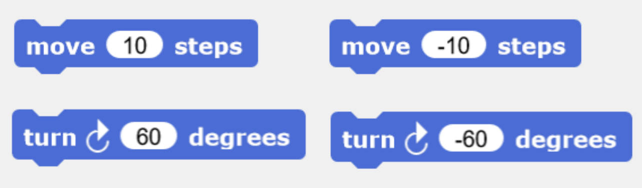
Maybe you confused x-coordinates and y-coordinates?



```
x position      y position
mouse x         mouse y
```

5.4 Mixing Up Numbers

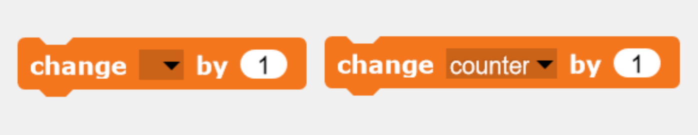
When you copy blocks and reuse them, you often need to adjust the numbers in there or change negative to positive numbers and vice versa:



```
move 10 steps              move -10 steps
turn ↻ 60 degrees          turn ↻ -60 degrees
```

5.5 Wrong Selection

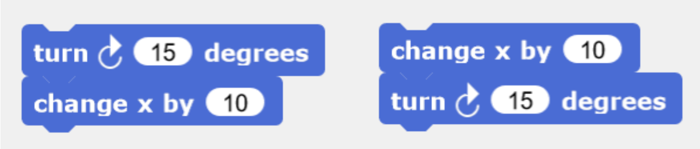
In some cases, the selection from dropdown menus is simply forgotten, see that you do not leave the dropdown menus empty:



```
change  by 1               change counter by 1
```

5.6 Wrong Order of Blocks

Are our blocks in the correct order? Sometimes even slight differences have can have large effects:

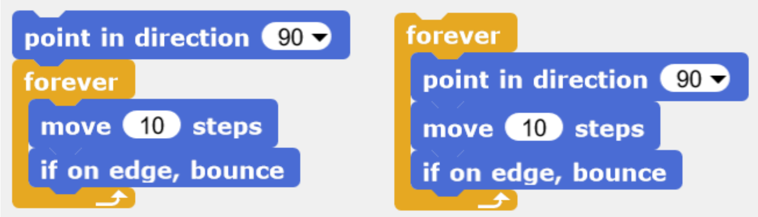


```
turn ↻ 15 degrees
change x by 10
```

```
change x by 10
turn ↻ 15 degrees
```

5.7 Loops

Are the blocks that need to be in the loop where they should be, or did you put them in the wrong place?

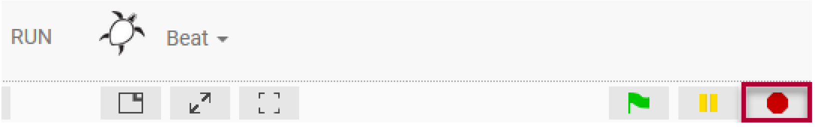


```
point in direction 90
forever
    move 10 steps
    if on edge, bounce
```

```
forever
    point in direction 90
    move 10 steps
    if on edge, bounce
```

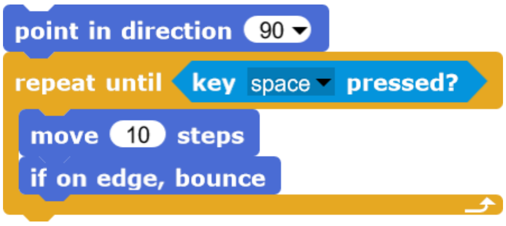
You see that these programs do different things, depending on where we put the “point in direction 90” block.

Moreover, you see that these programs both constitute an infinite loop, our turtle will be stitching tirelessly without an end. When this happens, click on the “Stop” above your stage to give your turtle a break and fix your code:



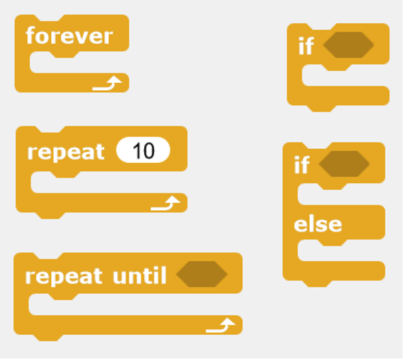
Your code needs condition in your loop that tells the turtle when to leave the loop.

In this example, our turtle will do what is in the loo repeatedly until some presses the spacebar. When this condition is met, the turtle will not start the next iteration of the loop and instead leave it.



```
point in direction 90
repeat until key space pressed?
    move 10 steps
    if on edge, bounce
```

In the above case, the loop is executed until an event happens. Something that is also very popular for loops is to let the code run for a specific number of iterations. The block "repeat 10" does this. However, it depends on the task at hand which loop or condition is the best for you. Try the different blocks and see what they to, and help each other understand the differences and decide for what they are good for:

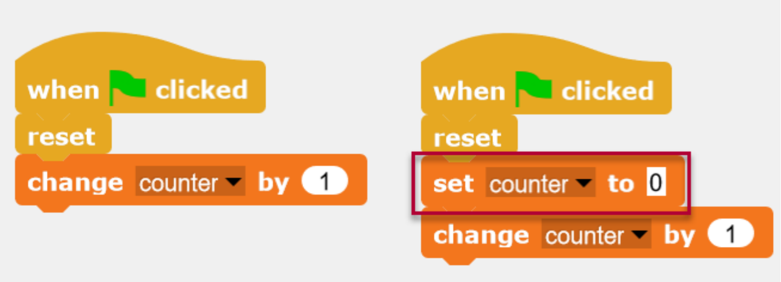


5.8 Initialize

When starting a program using the green flag, you may have to reset certain settings. This is also called initializing, for example when you reset your variables.

When you use a counter as in the program on the left, it is incremented each time the program is executed.

In case you want to make the counter start over upon program start, you need to tell it so, as in the example on the right where the counter is set to 0 after program start.



```
when ⚑ clicked
reset
change counter ▾ by 1
```

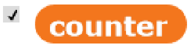
```
when ⚑ clicked
reset
set counter ▾ to 0
change counter ▾ by 1
```

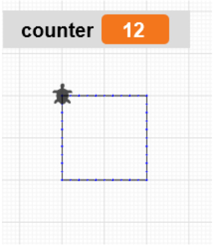
When your program does something else each time you start it, you probably forgot to initialize something in your code.

5.9 Show Values and States

To understand details of your code, it is sometimes necessary to show variable values on stage.

Check the box next to a variable in the palette to make it visible on stage. The current value will then be shown, and you can observe changes when you run your program. When you do not like that, simply remove the check from the box for that variable.

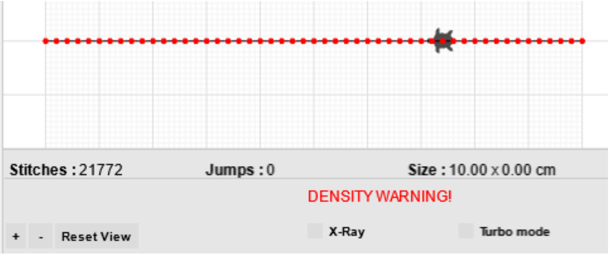




5.10 Density of Stitches

When you make too many stitches on the same spot, you will get a “Density Warning!” below your stage and the spot in question will be highlighted in red on your stage. These spots will be problematic when stitching, for example **the needle could get stuck, bend or even break. Please be cautious about this because you could injure yourself with a broken needle!**

Please to not stitch these examples with your embroidery machine, but first fix the code.

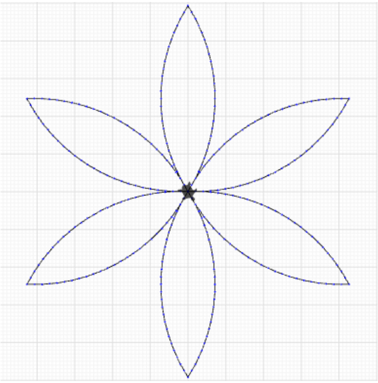


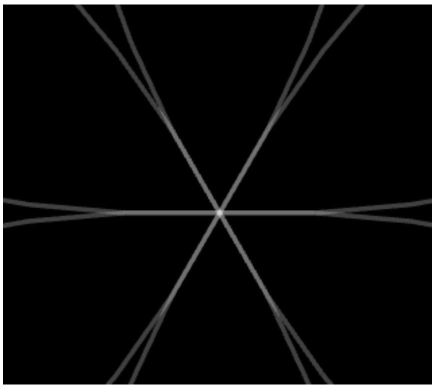
Stitches : 21772 Jumps : 0 Size : 10.00 x 0.00 cm

DENSITY WARNING!

Right below the “Density Warning!” you see the checkbox for “X-Ray”. Use this to show the threads in white. The brighter the white color, the denser your embroidery. If it gets too dense, the X-Ray mode also shows the spots in question in red as in normal mode.

This example flower looks OK in normal mode. Zoomed in and with X-Ray mode on, you see that certain areas overlap, and they are thus shown in brighter color. The center is the brightest spot as there are the most stitches, as each leaves starting point and end point is there, totaling in 12 stitches.

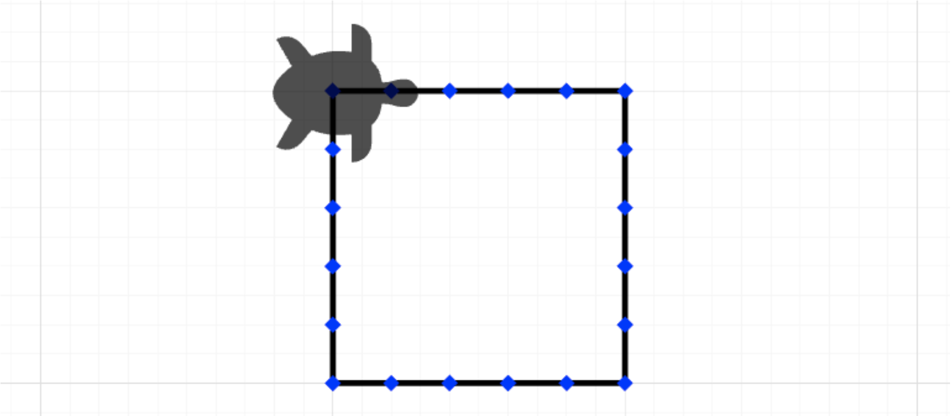




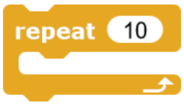
6 Examples

6.1 Square

We now stitch a square together. Follow the instructions below and create a copy of the code blocks shown.

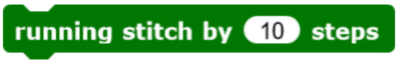


Blocks used:



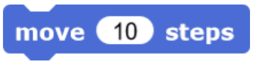
repeat 10

This block makes the embraced block sequence therein repeat the number of times that is indicated.



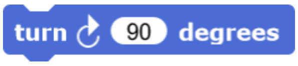
running stitch by 10 steps

This block defines the stitching settings we want to use (here it is a simple straight stitch, one step is 0.2 mm long).



move 10 steps

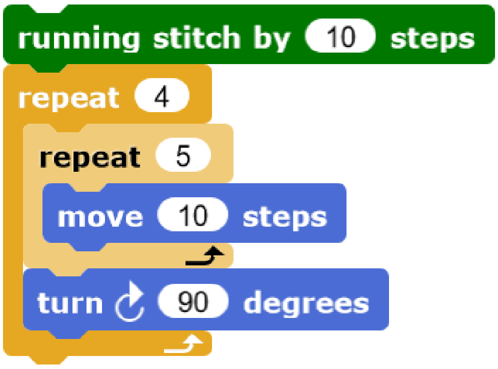
This block says how long the line to be stitched shall be. Again, one step means 0.2 mm.



turn ↻ 90 degrees

This block makes our turtle turn in the direction indicated and the number of degrees seen in the block. Note that there are different blocks for the two directions that you can turn.

Snap the blocks together and execute your code by clicking on it.



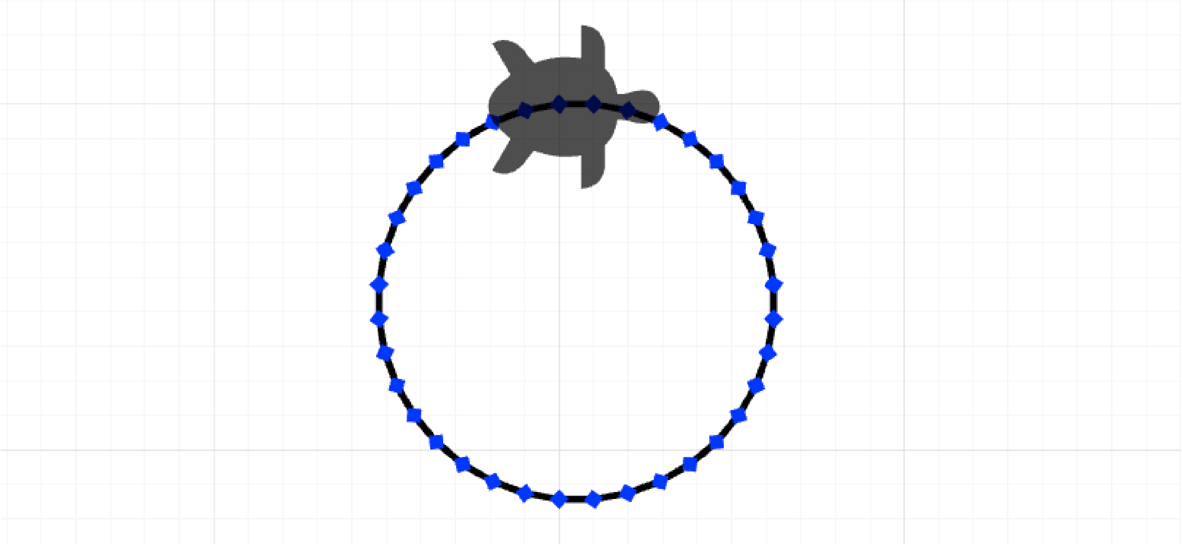
```
running stitch by (10) steps
repeat (4)
    repeat (5)
        move (10) steps
    turn ↻ (90) degrees
```

Now your turtle has stitched a square! Can you tell how large the square is and how many stitches there were required to create it?

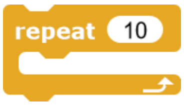
Solution: Each edge is 5 * 10 * 0.2 mm long. Thus, the square is 1 cm x 1 cm large. We needed 4 * 5 = 20 stitches to create the square.

6.2 Circle and Figures with Multiple Angles

We now stitch a circle together. Follow the instructions below and create a copy of the code blocks shown.

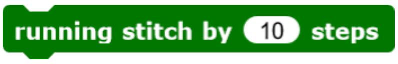


Blocks used:



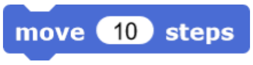
```
repeat (10)
```

This block makes the embraced block sequence therein repeat the number of times that is indicated.



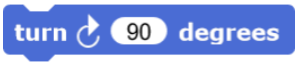
```
running stitch by (10) steps
```

This block defines the stitching settings we want to use (here it is a simple straight stitch, one step is 0.2 mm long).



```
move (10) steps
```

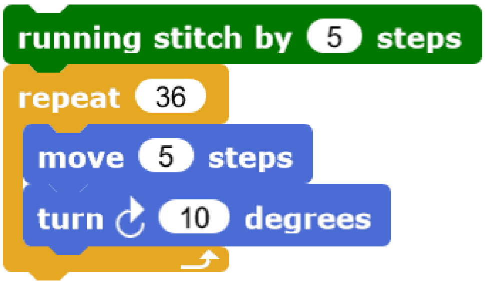
This block says how long the line to be stitched shall be. Again, one step means 0.2 mm.



```
turn ↻ (90) degrees
```

This block makes our turtle turn in the direction indicated and the number of degrees seen in the block. Note that there are different blocks for the two directions that you can turn.

Snap the blocks together and execute your code by clicking on it.



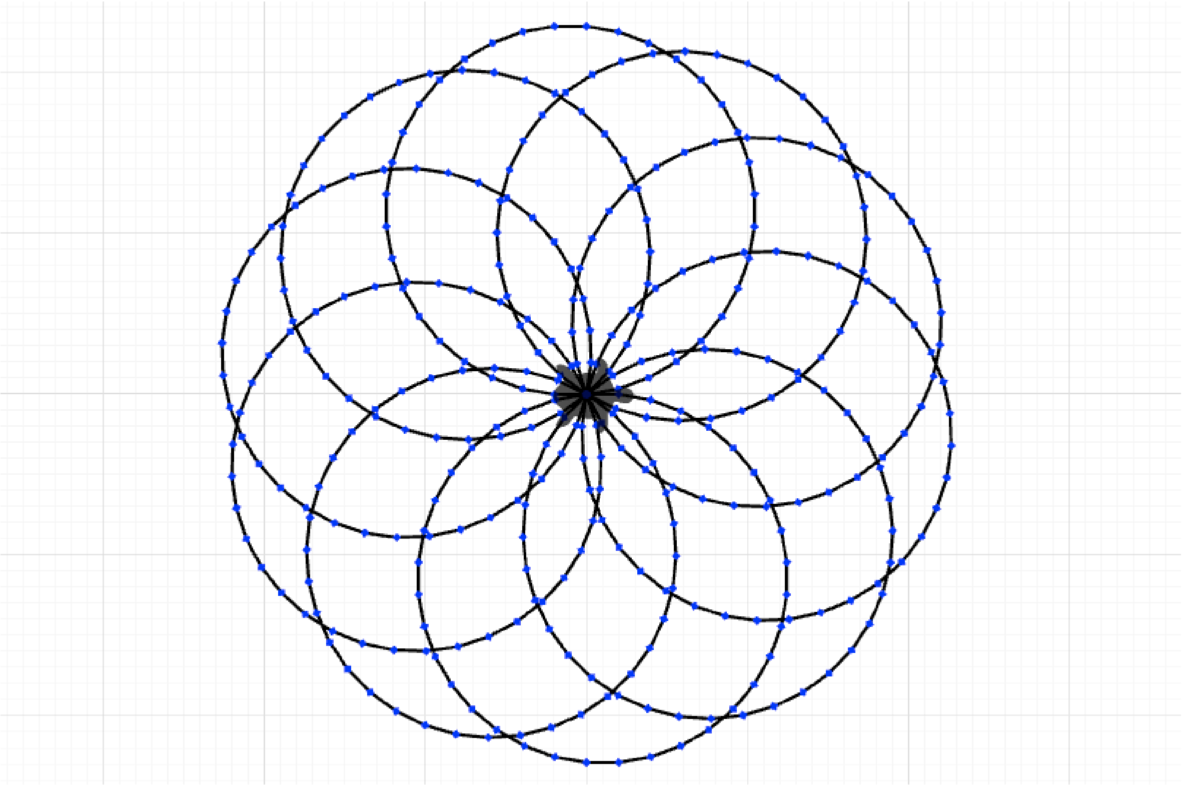
```
running stitch by (5) steps
repeat (36)
    move (5) steps
    turn ↻ (10) degrees
```

Now your turtle has stitched a circle! Or to be precise, it is a shape with 36 angles that looks like a circle. Try and change the values a bit and see if you can make the turtle stitch a hexagon (shape with 6 angles).

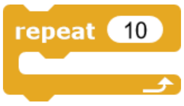
Solution: For stitching a hexagon, you need six edges, which means you repeat the straight line six times. In total, your turtle needs to turn 360° to get back to the starting point. For making six turns, you need to turn 60° for each turn to make a full turn.

6.3 Blossom

We now stitch a blossom together, which consists of several circles. Follow the instructions below and create a copy of the code blocks shown.

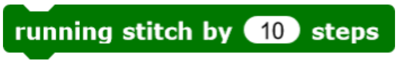


Blocks used:



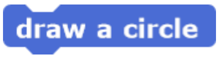
repeat 10

This block makes the embraced block sequence therein repeat the number of times that is indicated.



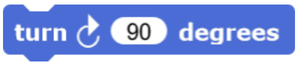
running stitch by 10 steps

This block defines the stitching settings we want to use (here it is a simple straight stitch, one step is 0.2 mm long).



draw a circle

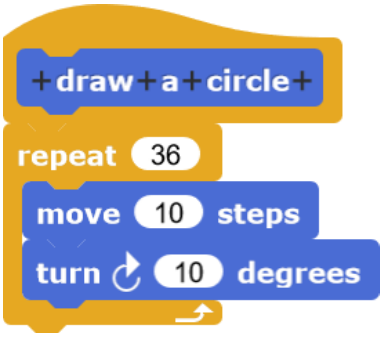
This is a personal block we created, which contains the program from chapter 6.2 “Circle and Figures with Multiple Angles” but without the stitching setting. See chapter 4.3 “Create Personal Coding Blocks” for explanation.



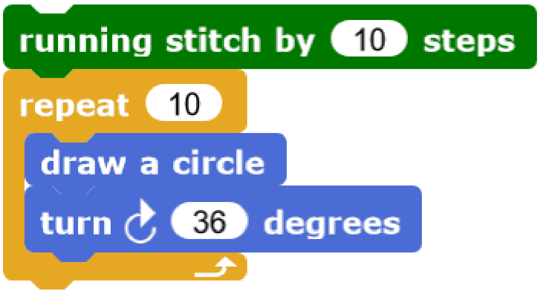
turn ↻ 90 degrees

This block makes our turtle turn in the direction indicated and the number of degrees seen in the block. Note that there are different blocks for the two directions that you can turn.

Snap the blocks together and execute your code by clicking on it.



```
+draw+a+circle+
repeat 36
    move 10 steps
    turn ↻ 10 degrees
```



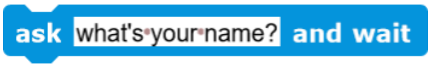
```
running stitch by 10 steps
repeat 10
    draw a circle
    turn ↻ 36 degrees
```

Now your turtle has stitched a blossom with ten leaves. How do you need to change the program for creating a blossom with 20 leaves? You will have a density problem here, but you can avoid it when you let the turtle move a couple of steps after every leaf. See chapter 5.10 “Density of Stitches” for details.

Solution: The number of leaves is determined by the iterations of circles drawn. To stitch 20 leaves, repeat the circle 20 times. Recalculate the angle that the turtle needs to turn after every circle. To come back to the starting point, the turtle needs to turn 360° and for 20 turns, this means 18° for each turn.

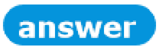
Now we want to change the program and let it calculate itself, what the angles after each circle need to be. The user only needs to select the number of leaves that the blossom shall have.

We will use the following blocks for this exercise:



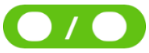
```
ask what's your name? and wait
```

This block asks the user a question. The stage will show the question and an area where the answer can be submitted. The program pauses until the answer is confirmed with the “Enter” key.



```
answer
```

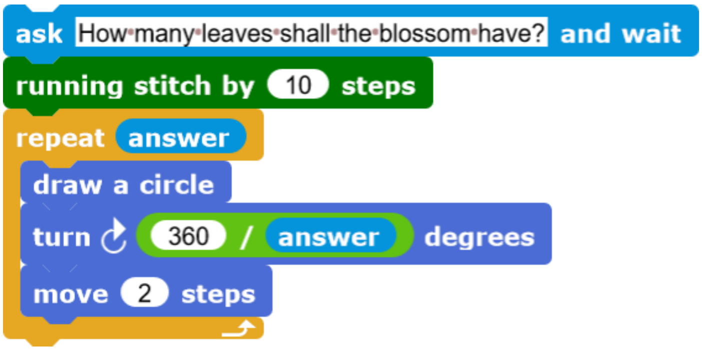
This block is a variable that save the answer submitted. The value remains until an answer for a new question is given.



```
( ) / ( )
```

This operator block calculates a division. When you click in the block, you can see the current result of the calculation that will be used by the program.

Snap the blocks together and execute your code by clicking on it.

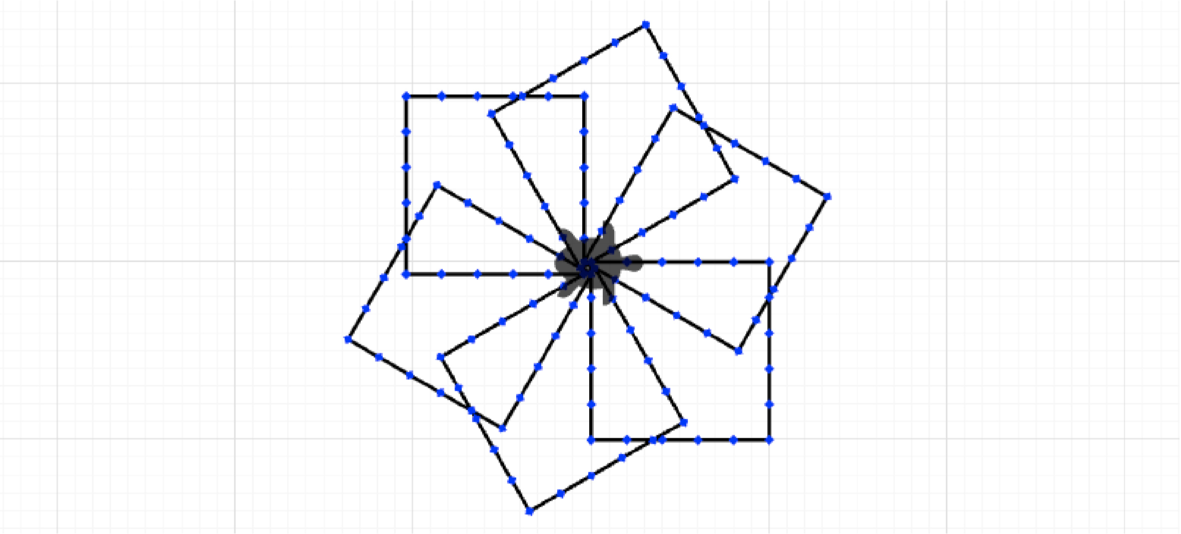


```
ask How many leaves shall the blossom have? and wait
running stitch by 10 steps
repeat answer
    draw a circle
    turn ↻ 360 / answer degrees
    move 2 steps
```

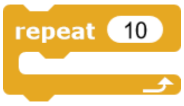
Now you can tell the turtle how many leaves it should stitch. Enter a number in the answer field for this.

6.4 Windmill

We now stitch a windmill together, which consists of several squares. Follow the instructions below and create a copy of the code blocks shown.

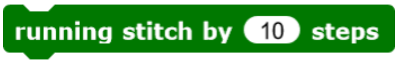


Blocks used:



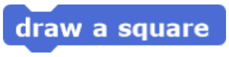
```
repeat 10
```

This block makes the embraced block sequence therein repeat the number of times that is indicated.



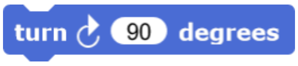
```
running stitch by 10 steps
```

This block defines the stitching settings we want to use (here it is a simple straight stitch, one step is 0.2 mm long).



```
draw a square
```

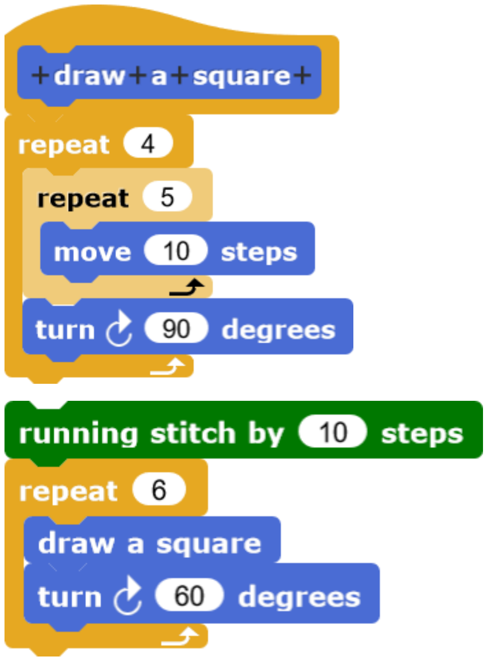
This is a personal block we created, which contains the program from chapter 6.1 “Square” but without the stitching setting. See chapter 4.3 “Create Personal Coding Blocks” for explanation.



```
turn ↻ 90 degrees
```

This block makes our turtle turn in the direction indicated and the number of degrees seen in the block. Note that there are different blocks for the two directions that you can turn.

Snap the blocks together and execute your code by clicking on it.



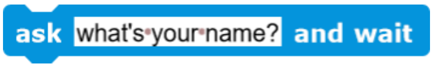
```
+draw+a+square+
repeat 4
    repeat 5
        move 10 steps
    turn ↻ 90 degrees

running stitch by 10 steps
repeat 6
    draw a square
    turn ↻ 60 degrees
```

Now your turtle has stitched a windmill with ten blades. How do you need to change the program for creating a windmill with 20 blades? You will have a density problem here, but you can avoid it when you let the turtle move a couple of steps after every leaf. See chapter 5.10 “Density of Stitches” for details.

Solution: The number of blades is determined by the iterations of squares drawn. To stitch 20 blades, repeat the square 20 times. Recalculate the angle that the turtle needs to turn after every square. To come back to the starting point, the turtle needs to turn 360° and for 20 turns, this means 18° for each turn.

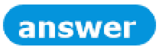
Now we want to change the program and let it calculate itself, what the angles after each square need to be. The user only needs to select the number of blades that the windmill shall have.

We will use the following blocks for this exercise:



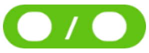
```
ask what's your name? and wait
```

This block asks the user a question. The stage will show the question and an area where the answer can be submitted. The program pauses until the answer is confirmed with the “Enter” key.



```
answer
```

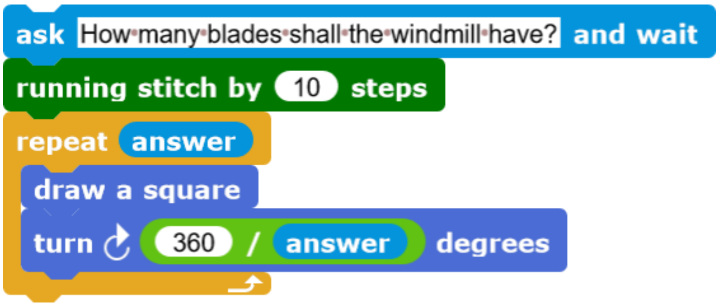
This block is a variable that save the answer submitted. The value remains until an answer for a new question is given.



```
( ) / ( )
```

This operator block calculates a division. When you click in the block, you can see the current result of the calculation that will be used by the program.

Snap the blocks together and execute your code by clicking on it.

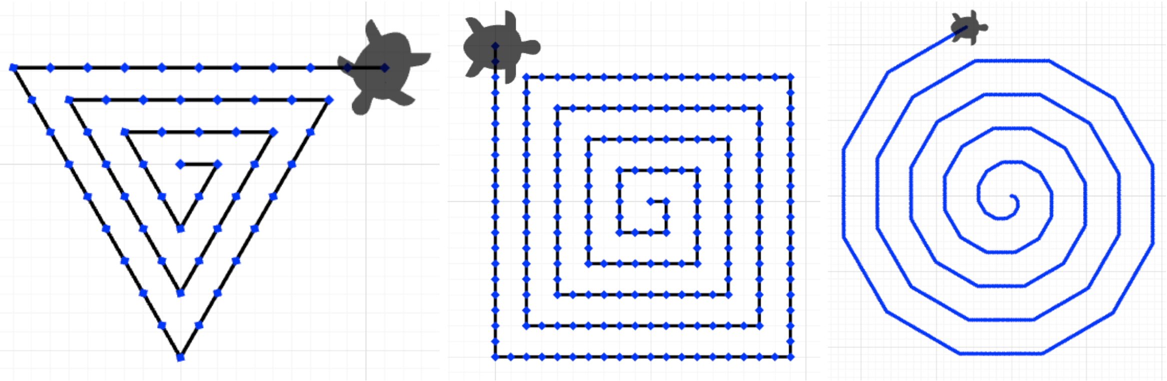


```
ask How many blades shall the windmill have? and wait
running stitch by 10 steps
repeat answer
    draw a square
    turn ↻ 360 / answer degrees
```

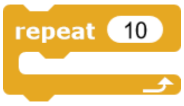
Now you can tell the turtle how many blades it should stitch. Enter a number in the answer field for this.

6.5 Spirals

Now we can stitch some spiral forms! Follow the instructions below and create a copy of the code blocks shown.

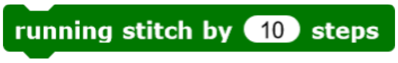


Blocks used:



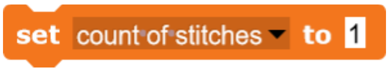
repeat 10

This block makes the embraced block sequence therein repeat the number of times that is indicated.



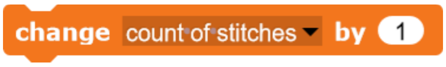
running stitch by 10 steps

This block defines the stitching settings we want to use (here it is a simple straight stitch, one step is 0.2 mm long).



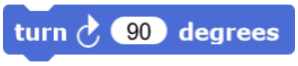
set count of stitches to 1

This block sets a previously defined variable to a certain value.



change count of stitches by 1

This block increments a previously set variable by 1. For decrementing the value, just let it add -1.

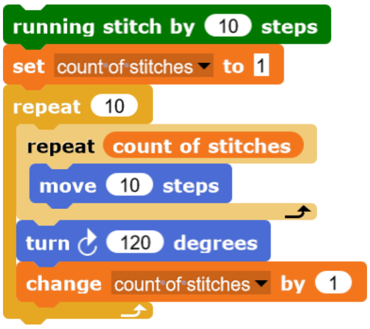


turn ↻ 90 degrees

This block makes our turtle turn in the direction indicated and the number of degrees seen in the block. Note that there are different blocks for the two directions that you can turn.

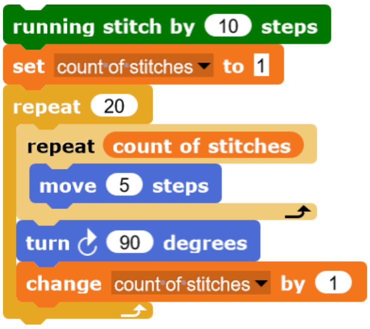
Snap the blocks together and execute your code by clicking on it.

Spiral with Triangles



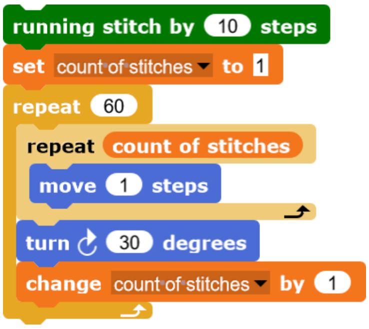
```
running stitch by (10) steps
set [count of stitches] to [1]
repeat (10)
    repeat (count of stitches)
        move (10) steps
    turn ↻ (120) degrees
    change [count of stitches] by (1)
```

Spiral with Squares



```
running stitch by (10) steps
set [count of stitches] to [1]
repeat (20)
    repeat (count of stitches)
        move (5) steps
    turn ↻ (90) degrees
    change [count of stitches] by (1)
```

Spiral



```
running stitch by (10) steps
set [count of stitches] to [1]
repeat (60)
    repeat (count of stitches)
        move (1) steps
    turn ↻ (30) degrees
    change [count of stitches] by (1)
```

We now created different spirals. But why does the last one get increasingly edgy?

> Solution: The further in the spiral, the smaller are the steps between the turns, so the steps are hardly visible. The further out in the spiral, the larger are the steps between the turns and are clearly visible, appearing as edges. To be precise, it is not a real spiral but a shape with many edges and turns.

Experiment with the values and such as number of iterations and angle sizes to create different spiral-looking shapes.

7 Using Embroidery Machine

7.1 Pattern Size

The pattern should not be larger than 18 cm height and 13 cm width. The stitching frame would probably be too small for larger patterns (adjust to your embroidery machine and equipment if necessary).

7.2 Export Pattern to USB-Stick

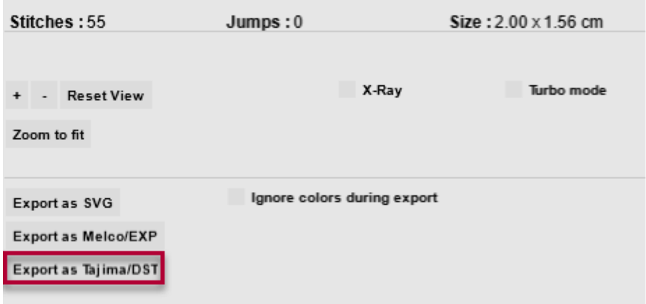
To stitch a pattern on your embroidery machine, you first need to export it from TurtleStitch to a USB-Stick. It is best to use an empty one.

You may create subfolders on your USB-Stick, but it depends on your stitching machine if and how many levels of subfolders it can read. With “Brother Innov-is 750E”, **only one level of folders and no subfolders** is allowed.

Do not use special characters for pattern names and folders on your USB-Stick.

To export your pattern, your code must be executed, and the pattern must be visible on the stage.

Choose “Export as Tajima/DST” from below the stage when using “Brother Innov-is 750E” or another method depending on your embroidery machine:



Stitches : 55 Jumps : 0 Size : 2.00 x 1.56 cm

☐ X-Ray ☐ Turbo mode

☐ Ignore colors during export

Choose where to save your file, namely the location on the USB-Stick and give it a name such as “my pattern.dst”.

How to use the USB-Stick to import and stick the file on the embroidery machine is described in the machine manual.

We wish you a lot of joy coding and creating and stitching your projects!

TURTLE STITCH

Goldau, 29. July 2021

Beat Horat (beat.horat@phsz.ch)

Mareen Przybylla (mareen.przybylla@phsz.ch)