LYNX

LCSI®

GETTING STARTED WITH LYNX

BASIC TECHNIQUES TO GET YOU STARTED

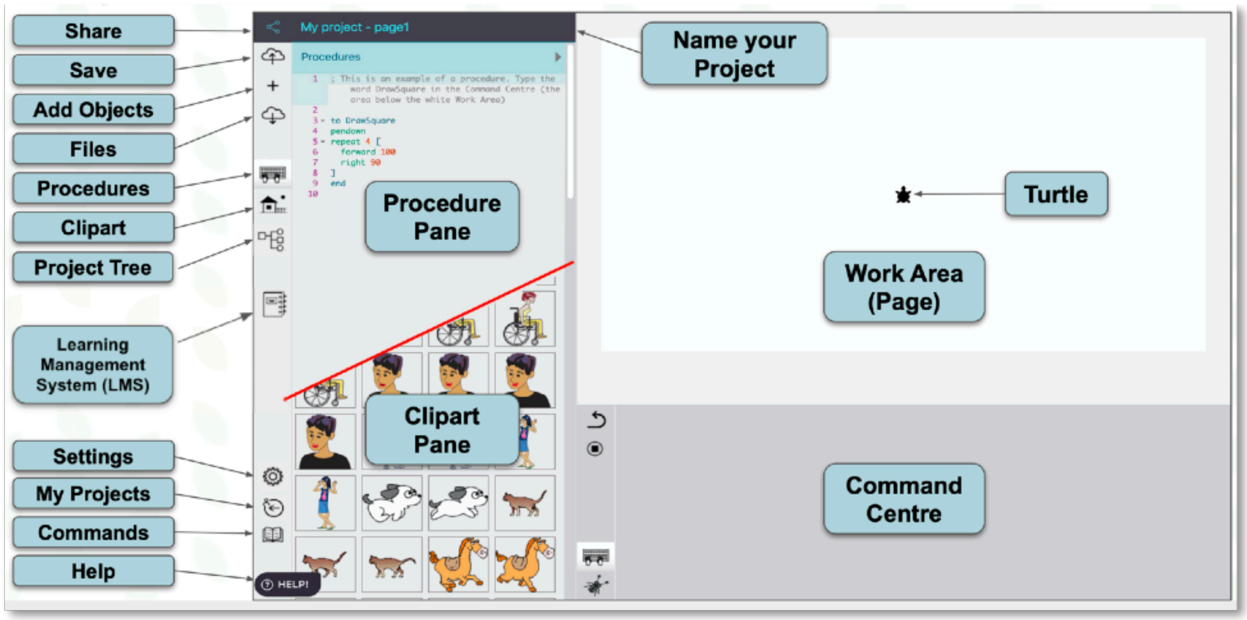
Table of Contents

Introduction

LYNX is a robust programming platform based on the Logo language. It is a more mature programming environment than coding with blocks, but it is a lot friendlier than professional languages like Javascript or Python. If you are somewhere between a block coder and a professional programmer, LYNX is for you.

You work on your LYNX projects in a browser window and you save them in the cloud. You can share your projects and even let your friends play or modify them.

The programming platform comprises a Work Area (you can have multiple pages in your project), a Command Center, a Procedures / Clipart / Project Tree Pane.



```
; This is an example of a procedure. Type the
  word DrawSquare in the Command Centre (the
  area below the white Work Area)

to DrawSquare
pendown
repeat 4 [
  forward 100
  right 90
]
end
```

BASIC CONCEPTS

Turtles: Turtles are everywhere in almost all LYNX projects. They obey your commands. They can draw, they can have different shapes, they can do animations, detect collisions and colors, and much more.

Primitives and procedures: LYNX is a language and a language has words. LYNX has a built-in vocabulary of more than 200 words, but you need just a few to start coding and have fun. The LYNX built-in vocabulary is called **Primitives**. Primitives are always present in every project, they always work. You can also create a **Procedure**, which is a group of primitives, to which you give a name. A Procedure adds a <u>new command</u> *only* for the project in which you created it. There is a section further down that explains how you can make your own Procedures.

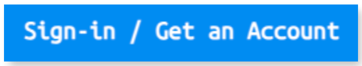
Command Center and Procedures Pane: The **Command Center** is a good place for trying things. Think of it as a *rough draft* area. The instructions you type in the Command Center are run *immediately*. This is also where error messages appear. Read them as they point to what the error is.

The **Procedure Pane** is where you define procedures. Procedures are *new commands* for your project. If you want to run the new commands:

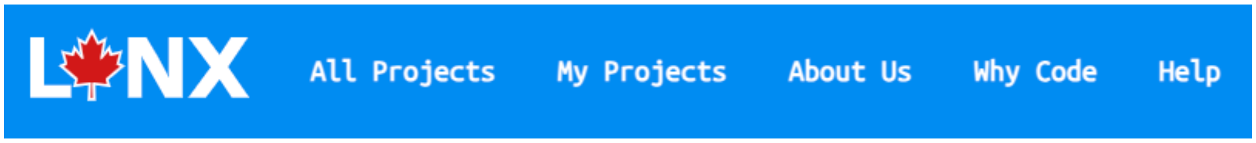
- Type the procedure name in the Command Center; or
- Add it to the On Click field inside a Button or
- Add it to the On Click field inside a Turtle Dialog Box.

Creating an Account and Creating your First Project

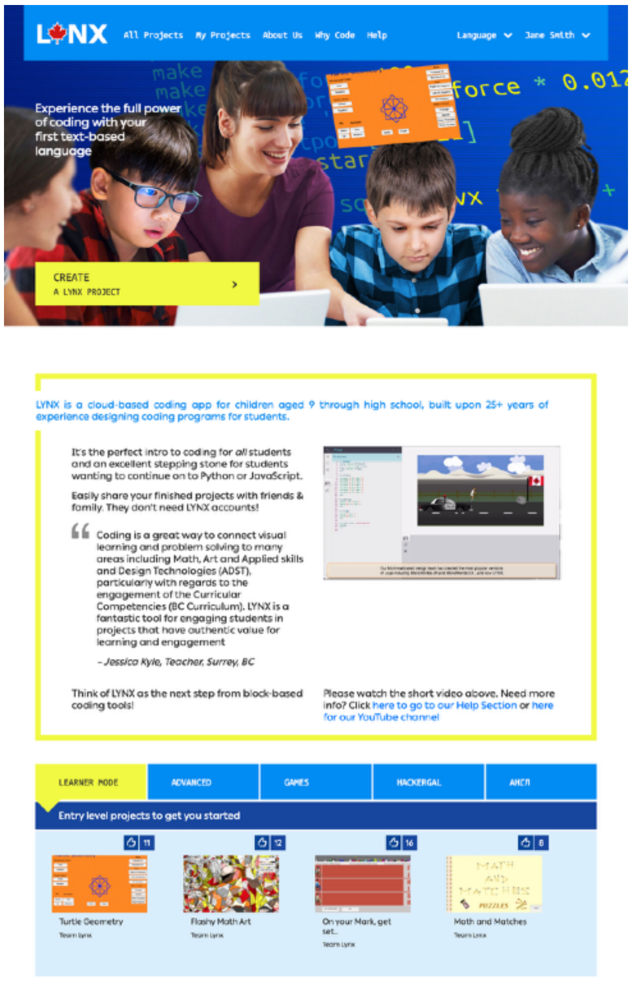
LYNX lets you create projects and also provides a space where your projects are saved. By default, it is a private space. You can *choose* to make your projects public or share them with others. More about this later.



Go to https://lynxcoding.club. If you see your **nickname** in the top-right corner, you are already registered and logged in. You can now go to **All Projects** or **My Projects** in the menu at the top of LYNX Home page or any other page of the LYNX web site.



If you see **Sign in / Get an Account**, click on it to create an account *or* sign-in if you have an account already. In the Help section of our website there is a PDF called How to Create a Free Trial LYNX Account *if you do not have one*. Follow the instructions in that PDF to create your account. We offer third-party account registration and sign-in using **Google** and **Microsoft**. **Once your account is created, click on Sign-in** and log in.



LYNX is a cloud-based coding app for children aged 9 through high school, built upon 25+ years of experience designing coding programs for students.

It's the perfect intro to coding for *all* students and an excellent stepping stone for students wanting to continue on to Python or JavaScript.

Easily share your finished projects with friends & family. They don't need LYNX accounts!

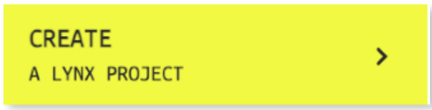
“ Coding is a great way to connect visual learning and problem solving to many areas including Math, Art and Applied skills and Design Technologies (ADST), particularly with regards to the engagement of the Curricular Competencies (BC Curriculum). LYNX is a fantastic tool for engaging students in projects that have authentic value for learning and engagement

– Jessica Kyle, Teacher, Surrey, BC

Think of LYNX as the next step from block-based coding tools!

Please watch the short video above. Need more info? Click here to go to our Help Section or here for our YouTube channel

Welcome aboard! On the left, you can see the LYNX home page. But after you sign in, you get directly to *your* private area in the LYNX cloud called **My Projects**. There, you will see projects that you have created (none if this is a new account), and a big yellow button allowing you to create your first LYNX project.



If this is your first time with LYNX, and in order to follow the instructions in the following pages, click on it.

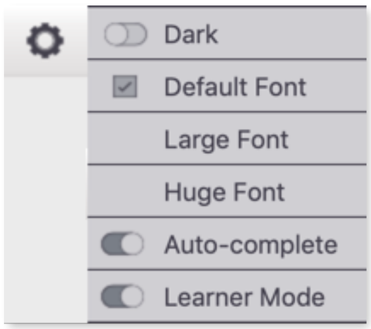
Note: You can only select the Work Area size when you start a new Project.

Explore LYNX's Settings

(UI color, font, auto-complete, learner mode…)

After clicking the Create a LYNX project button you'll find yourself in the *LYNX Editor*. It is the place where all the work on your projects happens. Start by customizing this environment. In the bottom left side of the Editor, click on the **Settings** icon:

From this menu, you can…


Dark
☑ Default Font
Large Font
Huge Font
Auto-complete
Learner Mode

1. Toggle **Dark mode** for a different UI look.

2. Choose one of three **font sizes** for the Procedures Pane and the Command Center.

3. Toggle the **Auto-complete** feature. LYNX suggests primitives as you type in the Procedures Pane or in the Command Center. Remember, a primitive is a command you use in LYNX. If you are new to LYNX, leave **Auto-complete** on.

4. Toggle **Learner Mode**. In Learner Mode, you'll get simplified Hints and Procedure Templates in the LYNX Editor. Also, Auto-completion will only suggest the 40 easiest and most popular primitives (commands).

The **Online Help Guide** icon is in the bottom left corner. This short Quick Reference Guide will show you explanations on how the most popular LYNX primitives work.

Also in the bottom left corner, you will see a Black Help widget. Click on it and type what info you are looking for in the *Search for Help* field. *Some* paid-for accounts will allow you to Contact Us and get the answer from a LYNX guru.

IMPORTANT! SAVE YOUR WORK OFTEN!

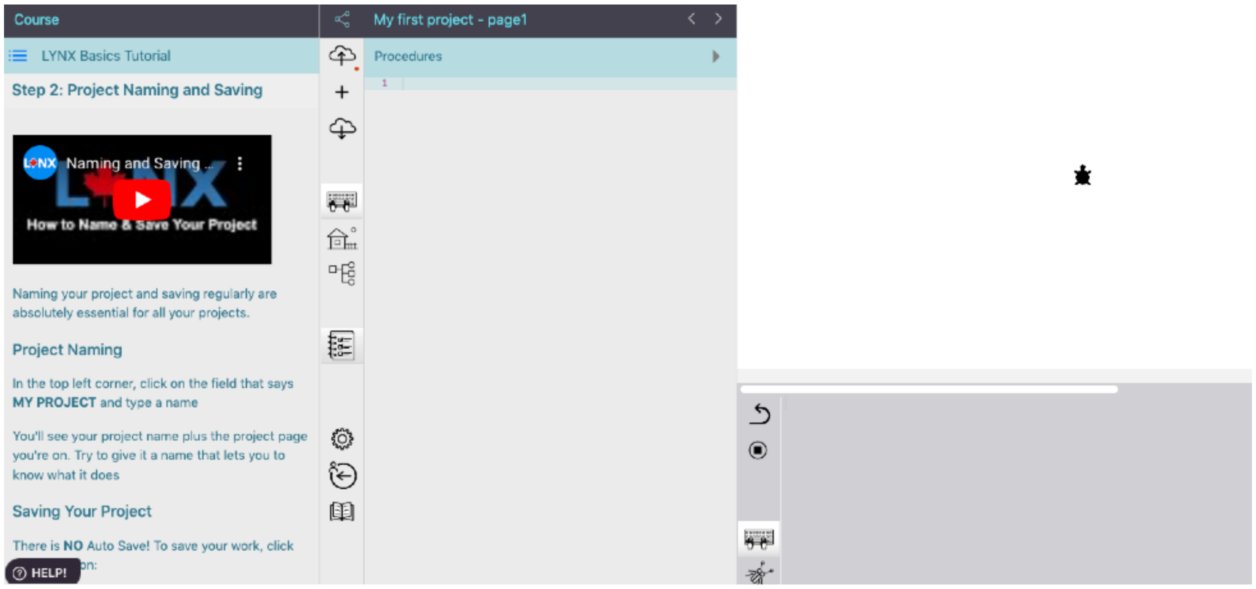
Please tell your students to save their project regularly. There is **NO** Autosave! To Save, go to the **Up to the Cloud** icon in the top left corner of the editor. There is a longer explanation about saving near the end of this PDF. The **red dot** is a warning that you have unsaved changes in your project.

Need Help Using LYNX?

LYNX will help you in many ways.

LEARNING MANAGEMENT SYSTEM (LMS)

The LMS should be visible when you open the Editor for the first time. You can open and close it using the spiral notepad icon on the left side.



Naming your project and saving regularly are absolutely essential for all your projects.

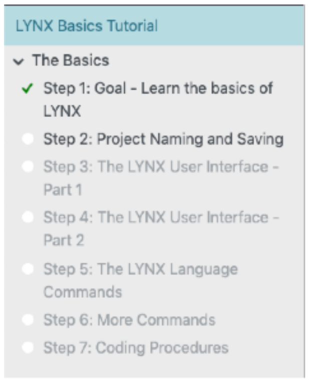
Project Naming

In the top left corner, click on the field that says **MY PROJECT** and type a name

You'll see your project name plus the project page you're on. Try to give it a name that lets you to know what it does

Saving Your Project

There is **NO** Auto Save! To save your work, click on:



LYNX Basics Tutorial

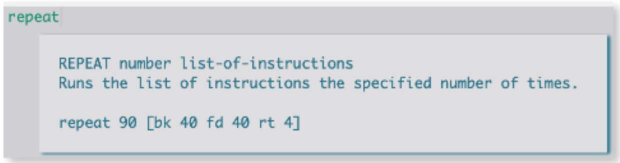
- The Basics
  - ✓ Step 1: Goal - Learn the basics of LYNX
  - Step 2: Project Naming and Saving
  - Step 3: The LYNX User Interface - Part 1
  - Step 4: The LYNX User Interface - Part 2
  - Step 5: The LYNX Language Commands
  - Step 6: More Commands
  - Step 7: Coding Procedures

Follow all the steps in the LMS. You can move forward or backward using the arrows at the bottom of each step. Make sure you open the YouTube videos and static pictures as you work through the steps.

The first LMS Tutorial, called LYNX Basics Tutorial, is seen on the left, and is available to *all* LYNX users, regardless of the type of LYNX account they have. There will be other Tutorials in LYNX but they may be available for specific LYNX accounts only.
FYI for teachers: teachers that have created a School / Club account can track the progress of their students in the LMS Tutorials that their School has subscribed to. In other words, you can see which steps each student has viewed.

TOOL TIPS

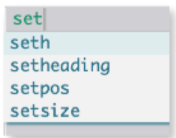
When you type a primitive in the Command Center or in the Procedures Pane, briefly leave your mouse pointer on the primitive and LYNX will display a floating tool tip with a short definition, the inputs required and an example of how it could be used.
The last page of this PDF has a list of the 40 most popular primitives.



```
repeat
REPEAT number list-of-instructions
Runs the list of instructions the specified number of times.

repeat 90 [bk 40 fd 40 rt 4]
```

Need Help Using LYNX? (cont.)

AUTO-COMPLETE



```
set
seth
setheading
setpos
setsize
```

When you start typing in the Command Center or in the Procedures Pane, LYNX suggests primitives that match what you are typing. Experienced LYNX users can turn this feature off in the **Settings** (gear) menu.

ONLINE HELP GUIDE

In the bottom left corner, click on the open book icon to see an **Online Help Guide** window. The Help pages match your current UI language and mode: Learner mode has a shorter Help. See an explanation of **Settings** earlier in this guide.

HELP WIDGET

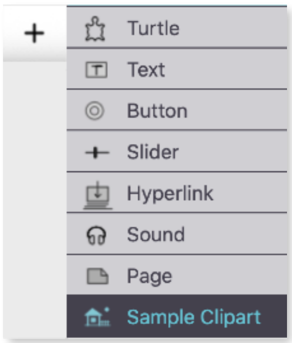
HELP!

The **Help Widget** is in the bottom left corner of the Editor. Click on it and type in a simple question, such as “How do I move the turtle”. A few explanations should appear; click on one. Only teachers that have a School or Club Account can email us a Question in the Last Resort *Contact Us* field. In other words, the *Contact Us feature* is *not* available for Trial and Individual accounts.

SAMPLE PROJECTS

Choose **Samples…** in the **Down From the Cloud** menu to see a list of samples that matches your mode: Learner mode has simpler samples. You can try the sample and read the explanations and comments written by us, in gray, in the Procedures Pane to understand how we created it. Use these samples for inspiration! You can modify our samples and save them as your own project.

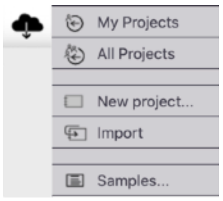
SAMPLE CLIPART



Want to try something quickly without finding some clipart on your own? Choose **Sample Clipart** in the “**+**” menu. Next, choose one or more of the categories and LYNX will populate the Clipart Pane with our clipart that you can use in your project. Later in this PDF, you can learn more about clipart.

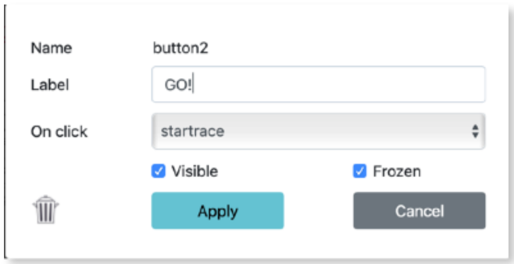
How to use the samples

Here's how you can use our samples to get ideas, learn tricks, "borrow" some code and just modify them to your heart's content.

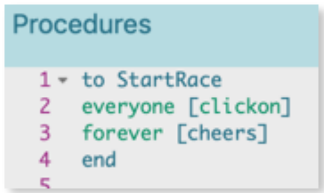


There are two sets of samples: Learner Mode (easy) and Advanced (challenging). Choose **Samples…** in the **Down From the Cloud** menu and choose one of the samples.

Play with the sample! Right-click * on buttons, turtles, text boxes, anything you find to see how they are made. Buttons and turtles often refer to procedures in the Procedures Pane, like this one: **button2** with the label GO! calls the procedure **startrace**:

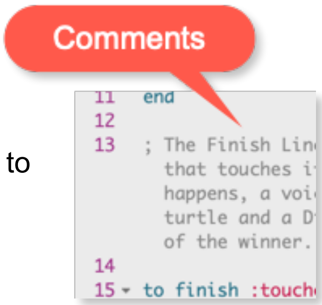


Name: button2
Label: GO!
On click: startrace
☑ Visible ☑ Frozen



Procedures

```
to StartRace
everyone [clickon]
forever [cheers]
end
```



```
11    end
12
13    ; The Finish Lin
        that touches i
        happens, a voi
        turtle and a D
        of the winner.
14
15    to finish :touch
```

Don't forget to read the comments in the Procedures Pane. When you see a semi-colon, these are *our* comments, *not part of the actual code*. You will learn a lot from reading comments and looking at the Procedure (usually immediately above or below it). And don't forget to add comments to your own code as well! Programmers use comments to explain, in clear language, what the code does. It's a good habit for coders!

You can try 3 different things with Samples:

- Make changes here and there, just to see if you understand what's going on.
- Copy some code, close the sample project and paste the code in your own project (you will certainly have to make adjustments. That's good!).
- Or make some changes and save the modified project as yours! Saving a project was explained on Page 5 and more Saving info is found near the end of this document.

* **Tablets**: In this document, "**clicking**" refers to computers. If you are using a tablet, take that as a "**touch**" gesture. Also, right-clicking is not possible on tablets. The section *The Project Tree*, on page 34, explains the alternate method for opening dialog boxes.

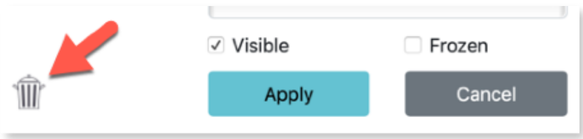
Adding, moving and turning turtles

ADDING TURTLES

In Learner Mode a new project comes with a turtle in the center of the page. If it's not there or if you want more turtles, simply choose **Turtle** in the "**+**" menu. Add a turtle now. If you already have a turtle in the center, the new turtle will appear *on top of it* so you may think that a second turtle has not been added. Drag the top turtle away to see both turtles. Turtles are named **t1**, **t2**, **t3** and so on, but, in a couple of pages, you'll see how to change the name of the turtle.

REMOVING TURTLES

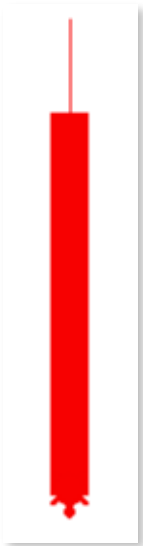
Want to remove a turtle? Right-click on a turtle to open its dialog box, then click on the trash can. Try it now and remove one turtle.



MOVING AND TURNING THE TURTLE, PEN UP AND PEN DOWN

Now that you have a turtle, type these commands in the Command Center (the gray area below the Work Area). After you press **Enter / Return** the code in that line will immediately execute.

```
forward 50
right 90
fd 100         (fd is short for forward)
rt 90          (rt is short for right)
pd             (means "pen down" - now the turtle will draw lines as it moves)
back 100       (or bk for short if you wish)
```



Now you know about `fd`, `bk`, `rt`, and `pd`. Experiment with these commands, so you can figure out what they do:

```
setcolor 'red' (Don't forget both apostrophes, setc is short for setcolor)
fd 50
setpensize 20
glide 200 1        (200 is the distance in pixels, 1 is the speed)
glide 200 10
pu                 (PU means "pen up" - now the turtle won't draw lines)
setsize 80         (the turtle doubles its size. FYI: setsize 40 is the normal size)
```

These instructions can also be put in procedures, and procedures can be called in many ways. Later in this guide are sections about creating procedures, buttons, clickable turtles, etc.

The color names for `setcolor` are: **black**, **blue**, **brown**, **cyan**, **gray**, **green**, **lime**, **magenta**, **orange**, **pink**, **red**, **sky**, **turquoise**, **violet**, **white**, **yellow**. Check all the color numbers and names at the end of this guide.

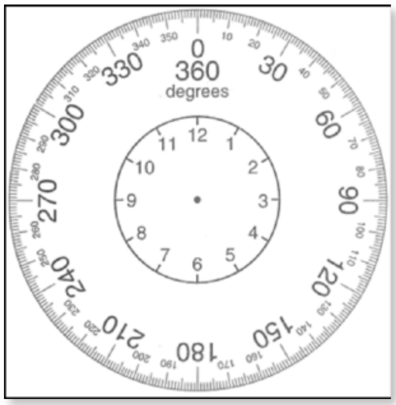
Adding, moving and turning turtles (cont.)

WHERE ARE YOU HEADING?

Besides right and left, there is another way to set the turtle's heading: `setheading` (or `seth` for short). Try these commands and see if you can spot the difference.

`cg` (stands for *clear graphics* - the work area is cleaned and the turtle is back in the center)

```
rt 90
rt 90
rt 90
cg
setheading 90
seth 90
seth 90
```



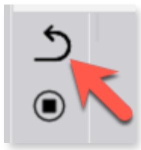
Each time you run `rt 90`, the turtle turns 90 degrees, *starting from its current heading*. It always turns 90 degrees **relative** to its current heading.

Each time you run `setheading 90`, you tell the turtle to "face East". You can tell it that several times, *it will always face East*. This is the **absolute** heading.

The input for `setheading` corresponds to the degrees on a compass.

OOPSIE!

If your last command (or last few commands) produces an unwanted result on your page, click on the **Undo** icon one or many times to undo them. This icon is just left of the Command Center.

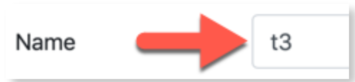


Adding, moving and turning turtles (cont.)

TALKING TO TURTLES (Hey! I'm talking to YOU!)

If you have only one turtle on the page, it will listen to every command you type or procedure you run. If you have *more than one turtle* on the page, *only one* will execute your commands: *it is the last one you created* ***or*** *the last one you clicked on.*

BUT… you can choose the turtle you want to run your commands. Say you have three turtles on the page, named t1, t2 and t3 (To see the turtle's name, right-click on each turtle and a dialog box will open. Look at the Name:



By knowing the turtle's name, you can "talk to" any turtle (or turtles), any time. (FYI: you can also type a different *one word name* for the turtle here). With t1, t2, and t3 on the page, you can use instructions such as these:

```
t1, forward 50
t2, glide 200 1
t3, setsize 80 wait 10 setsize 40
```
(get bigger, wait one second then smaller)

In the three lines above, don't forget the comma!

If you want to talk to several turtles at the same time, use the command `talkto`:

```
talkto [t1 t2]
setheading 90
```

Don't forget the square brackets after `talkto`. After you run `talkto`, all the turtles mentioned inside the square brackets will execute every command that you type in the Command Center until you address some other turtle(s).

There is also a way to talk to every turtle present on the page. The command is `everyone`.

```
everyone [setheading 180]
```

Unlike `talkto`, the primitive `everyone` only runs the instructions that you specifically mention in the square brackets. Use square brackets to indicate the instructions that every turtle on the page must execute.

For example, if you think you've *lost* a turtle, run:

`everyone [st]`   `st` means show turtle

Adding, moving and turning turtles (cont.)

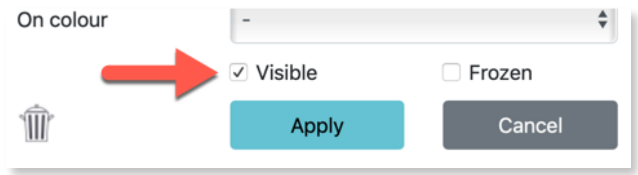
HIDING & SHOWING TURTLES

Turtles can be invisible! But before you hide a turtle, make sure you know its name, so you can see it again. Right-click a turtle and a dialog box will open. Remember the Name.

`t1, ht` (stands for hide turtle)

`t1, st` (stands for show turtle)

You can also use the **Visible** check box in the turtle's dialog box.



If you uncheck the **Visible** box, the turtle will no longer be visible.

If a turtle is hidden, you can also make it visible again by editing that turtle in the Project Tree. See the Project Tree section at the end of this Guide.

FREEZING TURTLES

Sometimes when you make a game you want to freeze the turtle so it's impossible to drag it around (to cheat 🙄). Try this:

`freeze 't1'` (that's the word *freeze*, a space, an apostrophe, the name of the turtle and another apostrophe)

Try to drag it around now. Good luck!

Now try this new command:

`unfreeze 't1'` (you can drag it around now)

You can also `freeze` and `unfreeze` a turtle from its dialog box. Look at the dialog box at the top of this page. Note that a frozen turtle will still execute your commands, for example `setsize` and `setc`… you just can't drag it around.

Inside the Project Tree, you can edit a Turtle and also unfreeze it.

Adding text

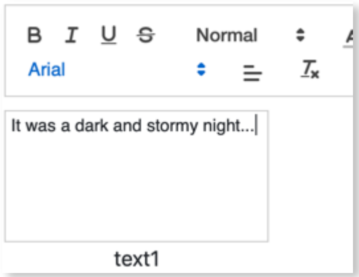
ADDING A TEXT BOX

You cannot type text directly in your Work Area, you must create a text box for that. Choose **Text** in the "**+**" menu, and a text box appears on the page. It is named **"text1"**, but you can rename it if you want.

TYPING TEXT USING THE KEYBOARD

You can just click inside the text box and start typing. Pretty obvious, right!

FORMATTING YOUR TEXT USING THE FORMATTING PALETTE



It was a dark and stormy night...

If you click inside a text box, above or below it you will see a box with formatting options. If you have selected text, the formatting applies to it. Otherwise, it sets the format for the text you are about to type at the cursor insertion point.

Type some text in a text box and try different formats.

If you can't see the formatting options, click on the text box name and drag the text box so that it's not so close to the edge of the Work Area.

TYPING TEXT USING COMMANDS

There are also commands that let you add text to the text box. Try these instructions in the Command Center. Remember that all these commands can also be part of procedures you make (more on that later on).

`print 'hello'`	Use an apostrophe to indicate that *hello* is just a word you wish to print, *not a command*. `Print` will print the text and brings the insertion point to the next line of the text box.
`print 'there'`	The insertion point should be on the next line. You can use `pr` for short.
`cleartext`	This empties the text box. You can also use `ct` for short.
`print [hello there]`	Use the square bracket to print several words at once.
`print 'hello there'`	Replace the brackets with apostrophes to get the same result.
`insert [My name is]`	`Insert` works like `print`, but the insertion point *stays on the same line*, ready to add more text on that line.
`insert 'Kim'`	Now *My name is Kim* should be visible in the text box as a single line of text,

MULTIPLE TEXT BOXES

If you have several text boxes on your page, you can use this method to talk to a specific text box. Don't forget the comma!

```
text1, print 'hello'
text2, print 'goodbye'
cleartext
```

The text box that "listens" to commands is:

- the last text box that you created; or
- the last text box that you used (clicked in); or
- the last text box that you "talked to" using the comma syntax described above.

In the example above, the `cleartext` command will be executed by the text box `text2`.

MOVING TEXT BOXES, VISIBLE, TRANSPARENT, WITH OR WITHOUT A LABEL

To move a text box, simply drag it by its label / name. In the example either `text1` or `text2`.

To make a text box bigger or smaller, in the bottom right corner of the text box, drag the arrow.

Right-click on the text box to open its dialog box. You can change its name – make sure you use a single word (with no spaces) for the name. You will see why later.

You can also hide or show the text box name.

Once you have written something in the text box, you can make the text box transparent so that the borders of the text box are removed.

Make the text box *invisible* by unchecking the Visible field in the dialog box *or* by typing `hidetext` in the Command Center. Making a text box *temporarily* invisible might be useful for storytelling, making a greeting card or for a game.

If you make the text box invisible, you can't open its dialog box again to make it visible. No worries, type this in the Command Center:

`showtext`	The text box reappears.

If you want to show or hide *another* text box, you will have to call it by its name like this:

`text1,`	Use its name, followed by a **comma**. This is why it is important to use a single word (no spaces) for naming things like text boxes. Now, this text box will listen to your commands.
`showtext`	`Hidetext` does the opposite.

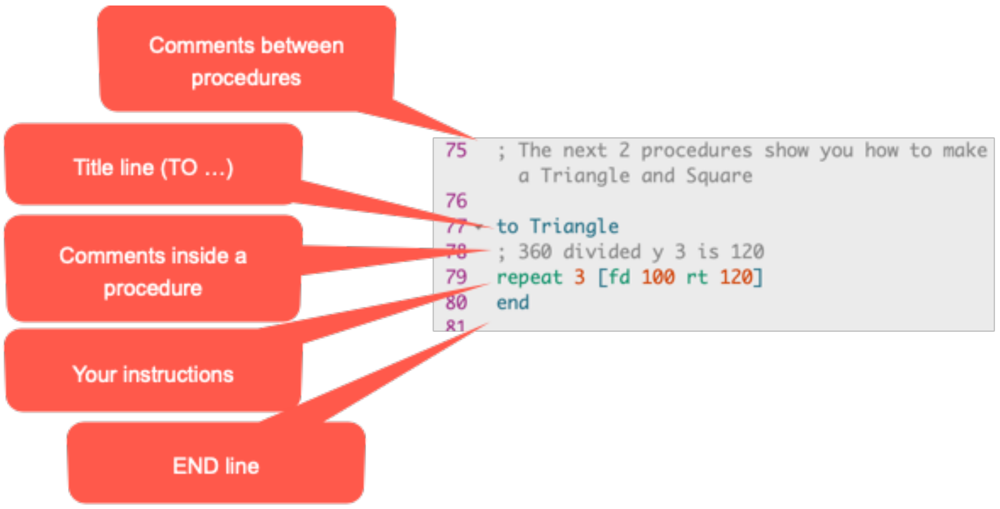
In the **Project Tree**, you can also edit a Text Box and make it visible again.

Creating procedures

You will see procedures all over this **Getting Started** guide. A procedure *adds a new command* that LYNX will understand but *only in the specific project you are working on*. Procedures not only help you organize your code in a logical manner but they make debugging easier.

Here are the first things you have to know about procedures:

- A procedure must start with the word `to`, followed by a space, and then the name of the procedure. The name must be *a single word with no spaces*. The name can't be a LYNX primitive like `forward` or `right` or a procedure that is already used in your project. These are good names: `Triangle` or `My_Triangle` or `MyGame`.
- Below the Title line, you can have one or many lines of instructions.
- Procedure instructions include primitives and even other procedures' names!
- A procedure must finish with the word `end`, all by itself on the last line.
- You can (and should) add comments before or even inside your procedures. All you have to do is start a line with a semi-column (;). Comments are in the color gray. Check the samples!



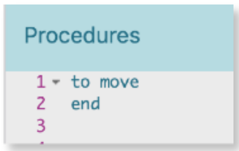
```
; The next 2 procedures show you how to make
  a Triangle and Square

to Triangle
; 360 divided y 3 is 120
repeat 3 [fd 100 rt 120]
end
```

- When a procedure is created, you can use it like any LYNX primitive:
  - Type the procedure name in the Command Center;
  - Add it to the dialog box of a turtle. See example on page 26;
  - Add it to the dialog box in a button (On Click field). See examples on pages 16 & 19;
  - Add it to another procedure.
- For color and collision detection, LYNX will prepare the procedure for you, as described later on.
- See Page 30 for more info on procedures.

Adding and using buttons

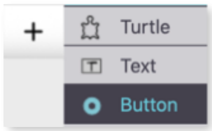
A button is *always* linked to a procedure that you created in the Procedures Pane. There are two ways to add a button and link it to a procedure.

OPTION 1: FIRST CREATE THE PROCEDURE



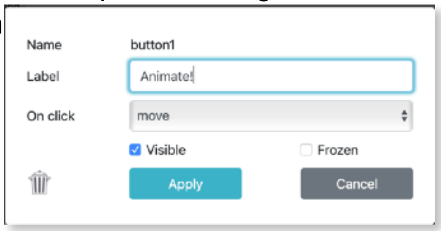
```
to move
end
```

This is the quickest method: start by creating a procedure. All procedures must start with to followed by a space, then a single word for example, `move`. Procedures must finish by the word end on the last line by itself. Not exactly sure how your procedure is going to work? Just create the title line and the end line.



Now create a button: choose **Button** in the "**+**" menu.

A button named **Nothing** appears on the page. Right-click on the button to open its dialog box. In the Label field, type something *meaningful* that you want to see on the button. The Label is plain English, not code, so it can be a few words. Then, in the **On Click** drop down menu, select the name of a procedure you have created. Click on Apply.

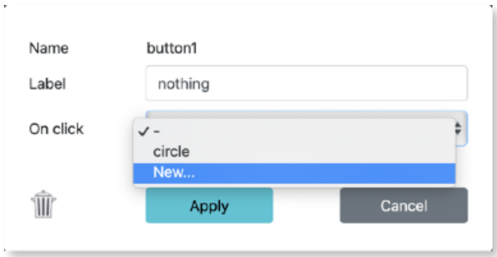


Name button1
Label Animate!
On click move
Visible [✓] Frozen [ ]

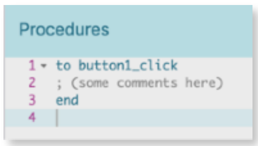
Now work on your `move` procedure and use your button to test it. If you want to change the size of the button, drag the arrow in its bottom right corner. You can move the button anywhere you like by dragging it.

OPTION 2: FIRST CREATE THE BUTTON

You can start by creating the button even if the procedure does not exist. Choose **Button** in the "**+**" menu. A button named **Nothing** appears on the page. Right-click on the button to open its dialog box. In the Label field, type something *meaningful* that you want to see on the button. Then, choose **New…** in the **On Click** drop down menu. Click on Apply. This creates a procedure named, for example, `button2_click` (because you already have a button from Option 1). LYNX will create a template of a procedure for you. Please note that it will be different depending on whether you are in Learner Mode or not. Use Learner Mode if you are just starting with LYNX.



Name button1
Label nothing
On click New...



```
to button1_click
; (some comments here)
end
```

Now work on your procedure. The second button is linked to it. Give your procedure a useful name. If your procedure makes the turtle bigger, you may want to call it Big_Size and add an instruction like `setsize 120`. If you rename your procedure (which is a good idea!), *you need to re-link the button to the procedure*.

Open the button's dialog box and choose your new procedure in the **On Click** drop down menu. In our Option 2 example, you would choose Big_Size.

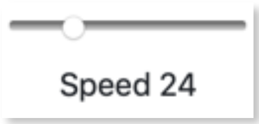
Adding and using sliders

CREATING A SLIDER

A slider is just like a variable number that you can control using your mouse. To create a slider, simply choose **Slider** in the "**+**" menu, and a slider appears on the page. It is named "**slider1**" - all one word. By default, the range of values go from 0 to 100 and the current value is 50. You can move the slider anywhere by dragging its name.

SET THE RANGE OF THE SLIDER

Right-click on the slider to open its dialog box. Here, you can change its **name** (always use a single word, with no spaces), its **min** and **max** values, and its **current** value (which must fall between the min and max values). You can also make it invisible, show or hide its name, and freeze it. When frozen, a slider can still be used but cannot be moved.



USING YOUR SLIDER (GET AND SET ITS VALUE)

The name of the slider reports its value. Type this in the Command Center:

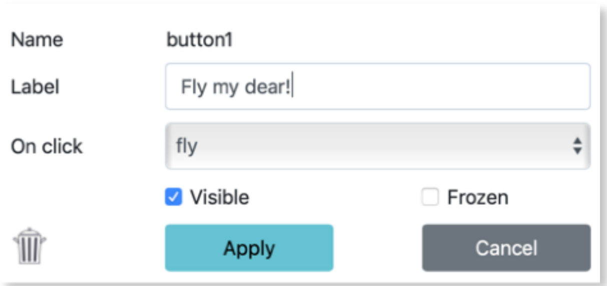
`show slider1`	Assuming you have a slider with that name on your page, this will print the current value of the slider in the Command Center.
`setslider1 10`	The word set, with the name of the slider (no space), will set the value of the slider without having to touch it.

USING YOUR SLIDER (TO CONTROL THINGS)

Make sure you have a turtle on the page, and set the slider range with a **min** value of **0** and a **max** value of **10**. Also change the name of the slider to **speed**.

Create a **fly** procedure like on the right side. The input to `forward` is the value of the slider called `speed`, divided by `10` (otherwise, the turtle would fly too fast).

```
to fly
forever [forward speed / 10]
end
```



Name	button1
Label	Fly my dear!
On click	fly
	[✓ Visible] Frozen

Create a button and set it to run the procedure **fly**. Click on the button, and use the slider to control the speed of the turtle.

To stop the turtle, click on the button again.

Adding sound and music

BRING A SOUND BITE INTO YOUR PROJECT

Remember to watch the YouTube video called **All about using Sound in LYNX**. You can find it in the **Help** Section in **Learning Key Techniques.**

The supported sound formats, using Chrome and Firefox, are: WAV, MP3 and M4A

To bring a sound file into your project, simply choose **Sound** in the "**+**" menu. The **Import Sound** dialog box appears. Choose a sound file already on your device. The Sound file you chose will now be visible in the URL field. It will probably be a long string of letters and numbers.

Click on **Create**. A sound icon appears on your page.

You can right-click on the sound icon to edit its name. Use a single word with no spaces. If you wish, you can also hide this icon or freeze it so it can't be moved anywhere.

USING SOUND IN YOUR PROJECT

You can click directly on the icon to play the sound, but sometimes, you will want to play the sound as your program runs (for example, when there is a collision or a winner to announce). The name of the sound clip is a command that plays the sound. Say you have a sound named **cheers** on your page (visible or not). You can type `cheers` in the Command Center, or include it as an instruction in a procedure.

Here is an example from our Get set, GO! sample. When the race starts the sound (`cheers`) begins.

```
to StartRace
everyone [clickon]
forever[cheers]
end
```

In the procedure above: `forever` will run the instruction (play the `cheers` sound) continuously, as an independent process. An independent process is run concurrently to other code that is running.

If you only want the `cheers` sound to play *once* at the start of the race, you can replace `forever` with `launch`.

If you want tons of sound effects that are free for educational purposes, go to this BBC site (BBC Sound effects): http://bbcsfx.acropolis.org.uk.

Adding and switching pages

ADDING A PAGE

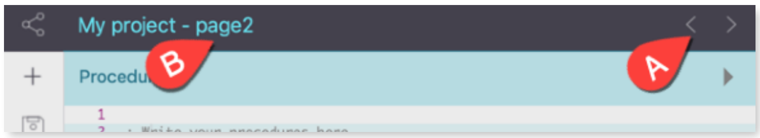
Remember to watch the YouTube video called **All about Pages in your LYNX Project**. You can find it in the **Help** Section in **Learning Key Techniques.**

One page is nice but you may want more pages if your project is a presentation, story, or a multi-level game, or if you want to write instructions for your project on a different page.

To add a page, simply choose **Page** in the "**+**" menu. You immediately get a new, empty page.

SWITCHING PAGES MANUALLY

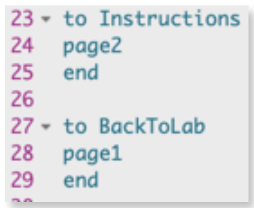
To switch pages manually, simply click on the left and right arrows at the top of the Procedures or Clipart Pane (A). The name of the page appears directly to the right of the project name (B):



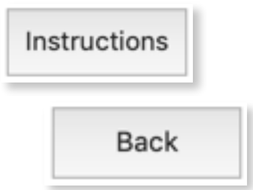
SWITCHING PAGES UNDER PROGRAM CONTROL

The name of a page is also a command that goes to that page. Here is an example in which a button on **Page1** leads to the Instructions page, and a button on **Page2** returns to the Lab page.

Create procedures such as these:



```
to Instructions
page2
end

to BackToLab
page1
end
```



Then, create a button on **Page1** that runs the procedure named `Instructions`, and a button on **Page2** that runs the procedure named `BackToLab`.

You can use page names just like as any other commands -- in a clickable turtle, in a collision detection or in a color detection. You can place a turtle near the bottom of your work area and, in the Sample Clipart, there are many shapes of arrows to use for turning a page.

RENAMING A PAGE

This is how you **rename** a page in your project, for example, from Page1 to Intro.

`rename 'page1' 'intro'`

You can also rename a page using the Edit button in the Project Tree. See more on page 34.

REMOVE A PAGE

To remove a page in your project, go to the Project Tree, click on the page you want to remove and click on **Delete** *or* type this in the Command Center:
`remove 'page3'` and press Enter / Return -- but be careful, this action cannot be undone!

Sample clipart

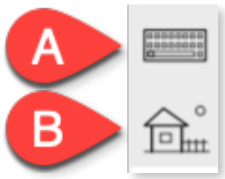
INTRODUCING THE CLIPART PANE

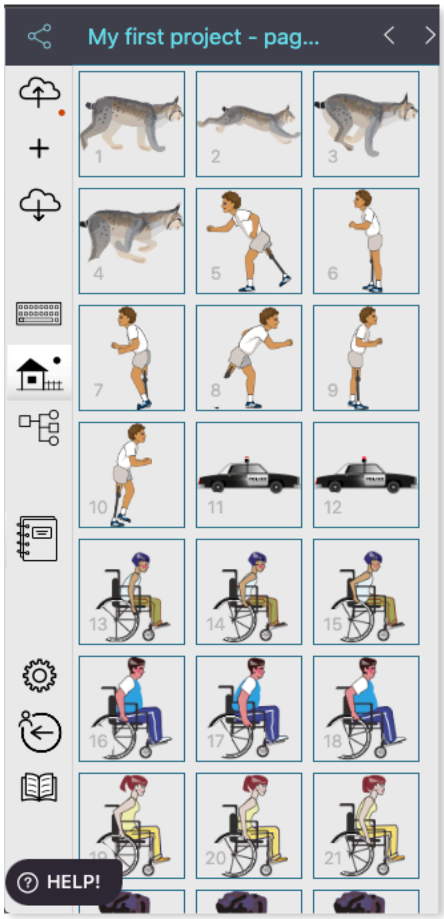
Remember to watch the YouTube video called **All about Clipart in LYNX**. You can find it in the **Help** Section in **Learning Key Techniques.**

Most projects need some sort of clipart, either as a full background image, as shapes for your turtles or as some other graphical elements.

A background image is simply a large clipart. Turtle shapes and graphical elements (a cloud for example) will be smaller. The sample clipart will help get you started quickly if you just want to play around with LYNX. On the next page, you will learn how to get your own clipart.

Click on the **House** icon (B) to open the **Clipart Pane**
(the **Keyboard** icon (A) will bring you back to the **Procedures Pane**).





ADDING OUR SAMPLE CLIPART

The Clipart Pane is now open and you see many empty boxes. Next, choose **Sample Clipart** in the “**+**” menu. There are 6 clipart categories to choose from. For example, choose **Backgrounds** and the **Clipart Pane** will be populated with very large clipart to use as backgrounds. You can choose one, or more, of the other 5 categories, if you like. The image on the left shows **Animations Clipart.**

The Clipart Pane can include *both* the Sample Clipart *and* your own clipart. It is the Clipart storage unit! There are 128 boxes.

See ***Adding your own clipart*** and ***Using clipart...*** further down.

Adding your own clipart

WHERE DO YOU GET CLIPART?

You can get your own clipart from three places:

- Use any paint program and draw your own images. Save your art as a JPG or PNG (PNG supports transparency around your objects if you need it).
- Download images from any image search web site. Respect copyright issues!
- Use the camera on your smartphone or tablet.

In any case, you may want to process your image before importing it into LYNX, so it has the right size (see below) or maybe for trimming the edges around objects to make them transparent (PNG only).

HOW TO IMPORT CLIPART (COPY-PASTE)

This is the *easiest* method. First, **copy some clipart** in a paint program of your choice or on a web page or by doing a screen capture. Press **Ctrl-C** if you are on Windows or Chromebooks (**Command-C** on a Mac).

Now click on the **House** icon to open the Clipart Pane. Click inside an empty clipart box to reveal a "**+**" sign. Now press **Ctrl-V** if you are on Windows or Chromebooks(**Command-V** on a Mac).

You can use a spot that is not empty if you do not want to keep the existing clipart. Click on the clipart, then click the **trash can**. Click again and you will get the "**+**".

HOW TO IMPORT CLIPART (IMPORT A FILE)

Click on the **House** icon to open the Clipart Pane. Click on an empty box to reveal a "**+**" sign in the bottom right corner, and click on that "**+**". Then use the dialog box to locate a clipart file on your device or online. Finish by clicking the Create button.

LARGE CLIPART FOR BACKGROUNDS

Run this instruction in the Command Center:

```
show projectsize
```

This will print a pair of numbers such as 800 450 in the Command Center. This is the **width** and **height** of your project in pixels, or turtle steps. Your numbers may be different.

When you search for images to be used as backgrounds, keep these numbers in mind.

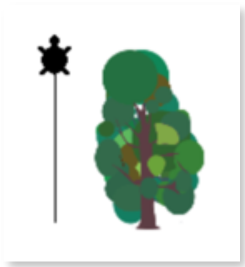
Too small: If you find images that are too small, you will have to stretch them to cover the entire background, and they will *not* look good.

Too large: If you find images that are way too large, that's OK, because as you import them, LYNX will resize them to match the project size.

Later on you will see you how to use your newly acquired clipart.

SMALLER CLIPART TO USE AS TURTLE SHAPES OR GRAPHICAL ELEMENTS

Same technique, but here, if you just need a cat, a ball, a tree or an atom, try to find images that are small relative to your project size. In this example, the tree is a 100-pixel clipart. How much is 100 pixels? Add a turtle to the work area and run `pendown forward 100`. That's 100 pixels.

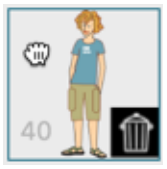


You can use `setsize` to resize the turtle, but there is no point having 5-pixel or 1,000-pixel clipart for turtles. So do the math. If your project is 800 x 450 (so 450 pixels vertically), and you need an object that's half of that, any image around 225 pixels is good.

Using clipart as turtle shapes

Assuming you added the Sample Clipart or imported your own clipart, now add a turtle to the page. There are 2 options to use to add a single Clipart as a costume / shape for the turtle.

Option 1: Easiest Way



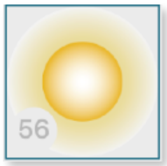
Choose a clipart image you like and click on that box.

A **white glove** appears. Move the white glove over to a Turtle and click directly on the turtle. The turtle should now be wearing the new clipart image.

Option 2: Using Commands

In the Clipart Pane, the clipart in each box has a number in the lower left corner.

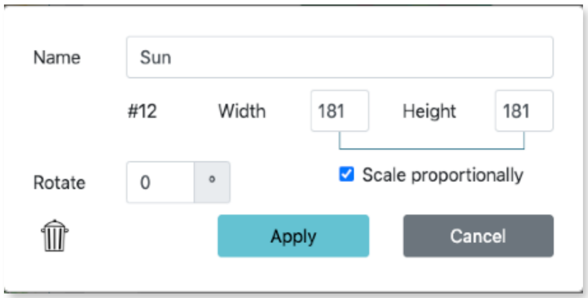
Choose a clipart you like and in the Command Center, type these instructions



`setshape 56`	Or use the number you chose. You can use `setsh` for short.
`setsh 0`	Want the original turtle shape back? Use 0.

More about Clipart

Right-click on a clipart image that is inside the Clipart Pane. You can give it a Name in its dialog box. Use a one-word name, with no spaces.



Name: Sun

#12 Width 181 Height 181

Rotate 0 °

☑ Scale proportionally

If **sun** is now the Name of the clipart image, you can type this in the Command Center:

`setsh 'sun'`	Use an apostrophe around *both* sides of the name, because **sun** is just a name, not a command that does something.

Note that only the normal turtle shape rotates (when you use `right`, `left`, or `setheading` commands). This means the head of the turtle will be pointing in the direction you chose.

Very important advice: Get your turtle moving in the direction you want *before* giving it a different shape! Make sure you like the direction it is going BEFORE adding clipart to it.

Using clipart as turtle shapes (cont.)

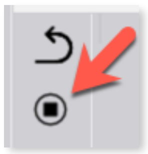
ANIMATION USING A SINGLE CLIPART

This example uses the Sample Clipart. It is a *really* good idea to test your setup and procedure using the normal turtle shape because you can see where the turtle is heading. Once you are happy with it, add a clipart shape.

Make sure you have a turtle on the page. Then type these instructions in the Command Center:

`setheading 90.` The head of the turtle now points to the right.

```
forever [forward 1 wait 1]
```



Click on the **Stop All** icon to stop the animation. It is the icon below the Undo icon just to the left of the Command Center. Change the values after `forward` and `wait` and see what happens.

Now type:

`setshape 11` The police car shape from the Sample Clipart. Your number may be different

`forever [forward __ wait __ ]` Use whatever numbers you like.

Remember to watch the YouTube video called **How to create a 1 frame animation**. You can find it in the **Help** Section in **Learning Key Techniques** or just click the preview on the right to play the video.

ANIMATION USING 2 OR MORE CLIPART

This example uses the Sample Clipart again with a Turtle moving to the right. These are shapes 38 and 39. Your numbers may be different. Try this:

```
setshape 0
setheading 90
setshape [38 39]
repeat 100 [forward 10 wait 3]
```

Here, `setshape` uses a list of numbers in square brackets as input. When you do that, LYNX switches shapes *each* time the turtle moves (`forward`, `back`). You should play with the numbers for `forward` and `wait` to create a realistic animation. Have fun!

Remember to watch the YouTube video called **How to create a 2 frame animation**. You can find it in the **Help** Section in **Learning Key Techniques** or just click the preview on the right to play the video.

Using clipart to make a nice background

Creating a background is a four-step process:

1. Import a large clipart or use our Sample Clipart, Backgrounds Category;
2. Give that Background shape to the turtle;
3. Stamp the turtle;
4. Set the turtle back to its original shape.

To create a nice background, you can use a paint program to draw an image, or you can just make a screen capture of something you like. Remember (see "**Getting your own clipart**" above) to use graphics that are about the size of your project or a bit larger.

Then import the file into an empty clipart box. This example uses the Sample Clipart, which has nice backgrounds. Type this in the Command Center:

`home`	Bring the turtle to the center of the page.
`setshape 17`	Use the number that corresponds to the background clipart spot you chose.

If the image does not fill the page, you can use commands such as:

```
setsize 60
```

You can also drag the turtle around to frame the image as you like. Yes, the image is a turtle with a huge shape. Now…

`stamp`	The turtle stamps its shape as an image on the background of the page. **However, the turtle is still present, with its huge shape.** Drag the turtle around. You will see a nice background fixed or stamped in the work area *and* a turtle that is moving around with the same background.

You probably want to get rid of the second background shape that moves with the turtle. Here is the solution:

`setshape 0`	Set the turtle back to its normal shape. Now you have a normal turtle and a stamped image in the background.

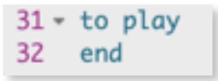
Remember to watch the YouTube video called **Adding Backgrounds**. You can find it in the **Help** Section in **Learning Key Techniques** or just click the preview on the right to play the video.

More about turtles: Making a clickable turtle

You can make something happen when a specific turtle is clicked on. A clickable turtle acts like a button. You can even add a clipart shape to make it look nicer.

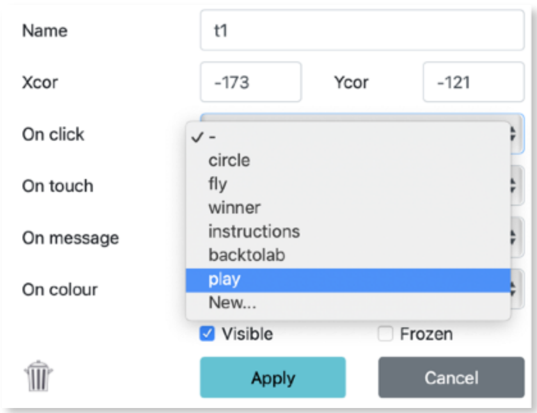
Making a clickable turtle is a three-step process:

1. Create a procedure for what the turtle should do when you click on it. If you are not sure yet, just create an empty procedure such as this:



```
to play
end
```

2. Add a turtle and right-click on it. In the turtle's dialog box, click on the **On Click** menu and choose the procedure that you just created, then click Apply to close the dialog box.



Name: t1
Xcor: -173 Ycor: -121
On click: play
On touch
On message
On colour

3. Now complete your `play` procedure by adding some instructions. Next, click on the turtle to test your procedure.

Different examples of what you can instruct the turtle to do with a simple click

Make a clickable turtle to turn pages.

Start by adding Sample Clipart, Arrows Buttons and give a colorful Arrow shape to the turtle. Follow the other 3 instructions directly above. Make sure you have a second page!

```
to continue
page2
end
```

Make a clickable turtle to start a game.

Note: this "GO!" button is an imported clipart shape.

```
to start
everyone [clickon]
end
```

Make a clickable turtle that displays instructions.

This "Question Mark" button is also an imported pic. Assuming that **text1** is the name of an existing text box on your page that contains the instructions:

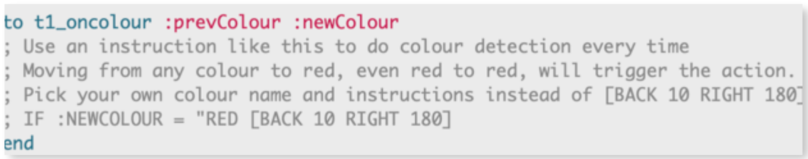
```
to instructions
text1, showtext wait 30 hidetext
end
```

More about turtles: Color detection (cont.)

You can use color detection when you want a turtle to do something when it moves and passes over a specific color. For example, use color detection to create a beep sound, force a bounce, go to the next page, declare a winner in a race, etc.

Programming a turtle for color detection is a multi-step process.

1. Add a turtle that's going to be moving - the one that is going to detect the color.
2. Tell the turtle that it is going to perform a color detection: open its dialog box, and in the On Color drop down menu choose New, and click on Apply.
3. Point 2 creates, in the Procedures Pane, a procedure such as this one:



```
to t1_oncolour :prevColour :newColour
; Use an instruction like this to do colour detection every time
; Moving from any colour to red, even red to red, will trigger the action.
; Pick your own colour name and instructions instead of [BACK 10 RIGHT 180]
; IF :NEWCOLOUR = "RED [BACK 10 RIGHT 180]
end
```

4. The gray lines starting with a semi-colon ";" are *just comments* describing the process. Read the comments and delete them if you wish. At this point, you want to know "what is the new color" that will trigger an event. Try this:
5. Draw a region and color it. For example:

`cg pu setx 200 pd fd 2000`	Clear the graphics, pen up, set the x coordinate to 200 (move to the right), put the pen down, and forward a lot to make a vertical line that splits the page in 2 sections.
`pu setx 210 setc 'red' fill`	Pen up, move the turtle a bit to the right, set the color to red and fill the area.

6. Next, you must place the turtle "*not on red*", and make sure it is pointing towards the red area. Type this in the Command Center:

`setc 'black'`	The Turtle is black again so you can see it.
`pu`	Pick the pen up.
`home`	Places the turtle at the 0 0 position.
`setheading 90`	Point towards the red area on the right.

7. Now you can make the turtle *do something* when it detects the color red. It's time to complete the procedure that was "prepared" in Steps 2 and 3 above. (In this example, the comments have been removed.):

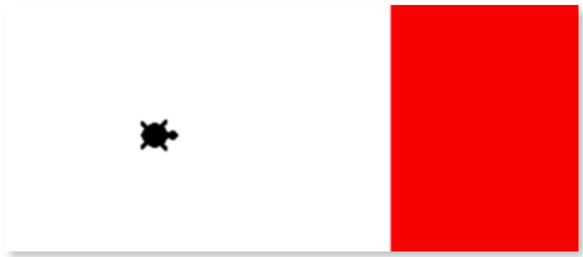
```
to t1_oncolor :prevColor :newColor
if :newcolor = 'red' [back 10 right 180]
end
```

8. Finally, get the turtle moving! Type this in the Command Center:

`forever [fd 2 wait 1]`	Moves the turtle so it hits the red wall.

More about turtles: Color detection (cont.)

The turtle will bounce when it touches the red area, then continue moving and bounce again when it touches red.



This example uses **red** as the color to be detected. These are other color names you can use: **black**, **blue**, **brown**, **cyan**, **gray**, **green**, **lime**, **magenta**, **orange**, **pink**, **red**, **sky**, **turquoise**, **violet**, **white**, **yellow**.

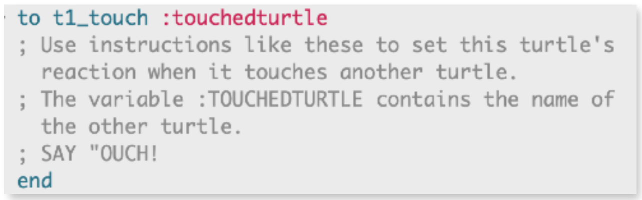
In short: if you are drawing a coloured area yourself, use `setcolor 'red'` (or any other color name) and `fill` to fill the area , and then use that *same color name* in your color detection procedure.

Important: If you are importing a background image, you should use a *solid color* for the color that the turtle will touch and detect.

More about turtles: Collision detection

Collision detection is similar to color detection, at least for the first steps:

1. Add a turtle that's going to be moving - the one that is going to touch another turtle.
2. Instruct it to detect a collision: right-click on it and open its dialog box and choose New in the **On Touch** drop down menu, and click on Apply.
3. In the Procedures Pane, you will see a procedure like the one on the right side.



```
to t1_touch :touchedturtle
; Use instructions like these to set this turtle's
  reaction when it touches another turtle.
; The variable :TOUCHEDTURTLE contains the name of
  the other turtle.
; SAY "OUCH!
end
```

4. The gray lines starting with a semi-colon ";" are our comments. The name of the procedure (**t1_touch**) means that this procedure applies to the turtle named t1, that's the one that has to be moving.
5. Edit the procedure that was "prepared" in Step 3 above:

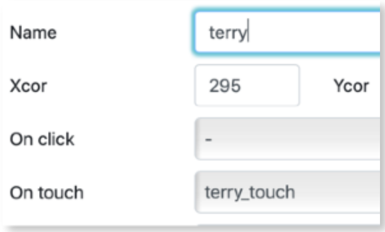
```
to t1_touch :touchedturtle
say 'hey'
end
```

6. Add another turtle to your project -- that is the turtle you want to touch / collide with. Now you can move or drag t1 so that it hits the second turtle. Did you hear 'hey'? If yes your **On Touch** procedure works! Make sure your computer volume is not set too low.
7. Your turtle can do different things when it touches different turtles. See the **:touchedturtle** variable in the Title Line? It lets you have different things happen, depending on "which turtle" is hit. It must always be there – even if you don't use it.

```
to t1_touch :touchedturtle
if :touchedturtle = 't2' [say 'Hey']
if :touchedturtle = 't3' [say 'Bye']
end
```

In the Terry Fox Learner Mode sample, Terry is obviously a turtle, and the rock, on the right hand side, is also a turtle named `rock`. Terry is programmed for this collision detection:

```
to terry_touch :touchedturtle
if :touchedturtle = 'rock' [stopall]
end
```

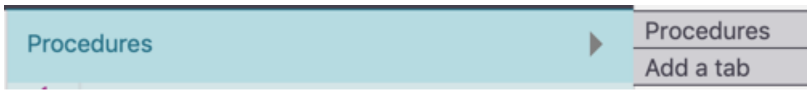


Name	terry		
Xcor	295	Ycor	
On click	-		
On touch	terry_touch		

More about procedures

You have seen procedures all over this PDF guide. You know almost everything about them, here's a recap:

- A procedure must start with the word `to`, followed by a space, and then the name of the procedure. The name must be a single word with *no* spaces. These are good names: `start` `TurnAround` `turn_around`.
- A procedure must always finish with the word `end`, *all by itself on the last line.*
- You can (and should) add comments before or even inside your procedures. All you have to do is start a line with a semi-column (;). Check the samples!
- A procedure may have an input, such as this one containing a variable:
  ```
  to square :size
  pd repeat 4 [fd :size rt 90]
  end
  ```
  In this case, you can't just use `square` as a command. Like `forward`, you HAVE to indicate "how much" the size should be. In the Command Center, use this procedure like this:
  ```
  square 50
  ```
  or
  ```
  square 100
  ```
- When a procedure is created, you can use it:
  - In the Command Center;
  - In a button;
  - In a clickable turtle; or
  - Even in another procedure!
- For color and collision detection, LYNX prepares the procedure for you, as described in previous pages.
- To better organize your procedures, you can create several Procedures tabs. Click on the arrow on the right side to create a new tab, then click on the arrow again to switch from tab to tab.
- A Procedure you have created will work *anywhere* in your current LYNX project no matter what page or Procedure tab it was created on. The order in which they were created is not important.
- You could think about organizing your procedures in a different tab *for each page* <u>and</u> giving each Procedures Tab a meaningful name.



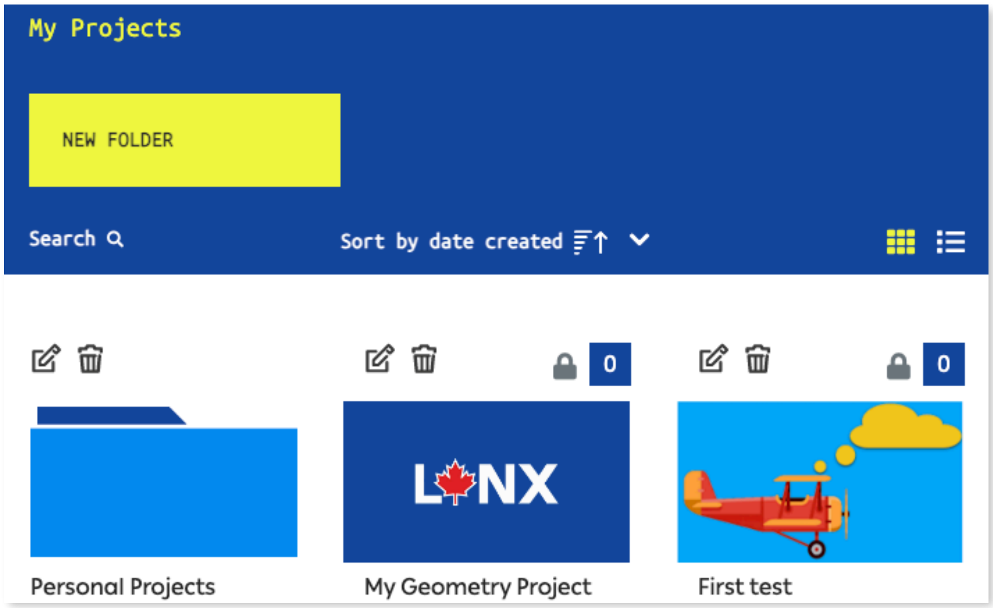
Saving and retrieving your project

Important: **There is no autosave,** not even when you leave your project to open another project *or* when you close the project *or when* you leave your browser window. *Save your project as you are working on it* and before leaving the LYNX project editor.

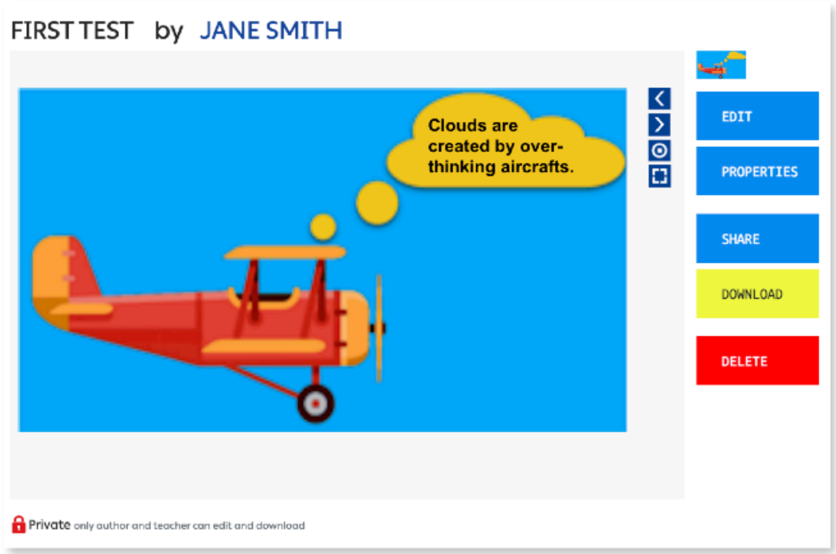
Saving is easy! You gave a name to your project when you created it. As you work, simply click on the **Up to the Cloud** icon.

A red dot near the **Up to the Cloud** icon means that your project has been modified since the last save - and you should *save it again* if these changes are worth saving.

When your project is saved, you can go back to **My projects** (create a bookmark for that page, that's your LYNX personal place in the cloud). Here is an example, with one folder and two projects. Projects have a generic "LYNX logo on blue background" image (see My Geometry Project below) unless you give them a customized Preview image (see First Test below)



When you want to work on your project again, go back to **My projects** and click on your project. Look inside a folder if you've put it there. You will see your project in **PLAY MODE**. If your project has buttons and clickable turtles, you can play with it right here.


FIRST TEST by JANE SMITH
Clouds are created by over-thinking aircrafts.
EDIT
PROPERTIES
SHARE
DOWNLOAD
DELETE
Private only author and teacher can edit and download

ABOUT THE BUTTONS ON THE RIGHT SIDE*

Click on **Edit** to open this project in the editor (and keep working on it).

Click on **Properties** if you want to change the name of the project, move it into a folder you have created, add a short description / tagline, set keywords, or give it a customized **Preview** image like you see above. To create a **Preview** image, make a screen capture of your project and save it on your computer. Then use the Browse button to upload it. By default, your project is saved as Private and will *not* appear in the **All Projects** section. Remember to scroll down and Save your changes.

Click on **Share** to obtain a link which you can copy and paste into an email or elsewhere.

Click on **Download** if you want to have a copy of this project on your computer. It will be called **ProjectTitle.js**, probably in your **Downloads** folder. Note: if you wish to send this project by email, be aware that *some* email servers will reject an attachment with a **js** extension. The **Share** button is a better way to share your projects.

Click on **Delete** to remove this project from your personal space ***forever!*** Careful !

* On smaller screens, these buttons may appear below your LYNX project.

Sharing your project with your friends

There are several ways to share your LYNX project:

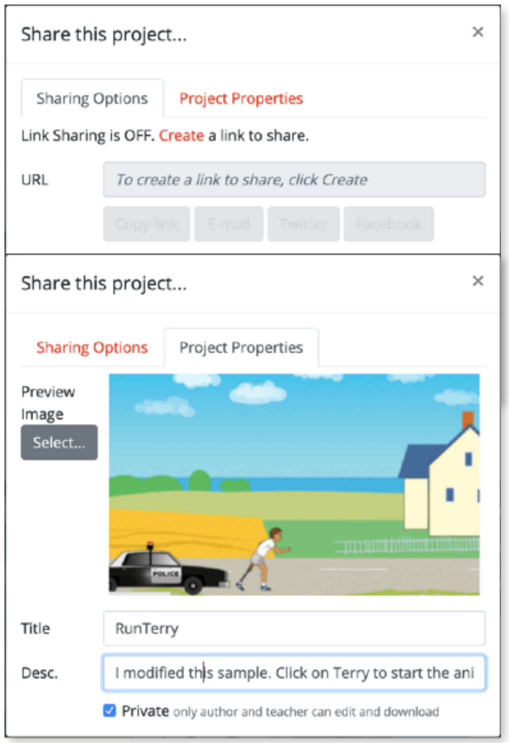
- from inside the LYNX editor, or
- from your LYNX personal space in the cloud (the **My Projects** page).

INSIDE THE LYNX EDITOR

To share a project from within the LYNX editor, simply click on the **Share** icon in the top-left corner of the editor.

The dialog box that appears has two tabs. In the first tab (**Sharing Options**), click on **create** a link to share the project, When the link is created, use the buttons directly below to **copy** the link to share by email or in other ways.

In the second tab (**Project Properties**), you can choose an image file to use as a Preview image, enter a title and a short description.



Share this project... ×

Sharing Options | Project Properties

Link Sharing is OFF. Create a link to share.

URL: To create a link to share, click Create

Copy link | E-mail | Twitter | Facebook

Share this project... ×

Sharing Options | Project Properties

Preview Image: Select...

Title: RunTerry

Desc.: I modified this sample. Click on Terry to start the ani

☑ Private only author and teacher can edit and download

IMPORTANT: Check the box Private to share your project while leaving it in your private “My Projects” area. The people with whom you share your project will be able to see and play with your project, and they can save a copy in their own LYNX space, if they have an account.

FROM WITHIN YOUR LYNX PERSONAL SPACE IN THE CLOUD

On your **My Projects** page, click on your project to open it in **PLAY MODE**. Then click on **Share** to get the same dialog box as described above.

ENJOY (AND EDIT) MY PROJECT

In the **Project Properties** tab of the Share dialog box, if you *uncheck* the Private box, people with whom you share your project will be able to see and play with your project *and* your project will also become visible in the **All Projects** section of the LYNX website. You can still share your project “personally” with your friends as described above, but you will also be sharing it with *anyone* in the world who visits the LYNX website.

Important: If your Project is now in the **All Projects** section of the LYNX website, it can be *modified* and *saved* by others but your personal project will always remain just as you created it.

The Project Tree

The project tree is a place where you can see and manage all the objects in your project. It is extremely useful in the following situations:

- To find a turtle or a text box that you have made invisible, and make it visible again.
- To manage all your objects in one place, without having to go from page to page. For example, to delete unwanted pages, turtles, text boxes…
- To edit your objects on a tablet, when a right-click is not possible.

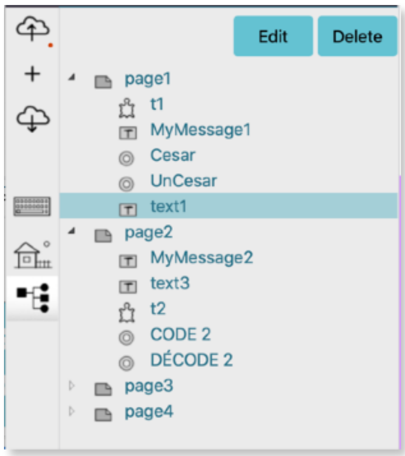
Click on this icon to open the project tree. The Project Tree appears in the Procedures / Clipart area.

The first thing you see in the Project Tree is the list of your pages. Click on the small triangle (page1) by the page name to expand the page and see the list of objects it contains. Click on it again to collapse the list.

When expanded, the page shows all the objects that page contains, even if the objects are invisible on the page.

Click on an object to select it, then click on **Edit** to open its dialog box, or click on **Delete** to delete the object. **Be careful! Deleting an object *cannot* be undone**.

In the example on the right, **text1** (text box 1) has been **selected** and can be **edited** or **permanently deleted** from the project.



POWERFUL FEATURE: COPY PAGES AND OBJECTS

The Project Tree is also a great way to copy *all the objects on a Page* (Turtles, Text Boxes, Buttons, Sliders, etc.) and paste them on another page.

Here's how to do this:
Step 1. Locate and Select the Page you want to Copy in the Project Tree.

Step 2. Press **Ctrl-C** (**Command-C** on a Mac).

Step 3. Now press **Ctrl-V** (**Command-V** on a Mac). You should now have an identical copy, *on a new page*, of whatever the original page contained.

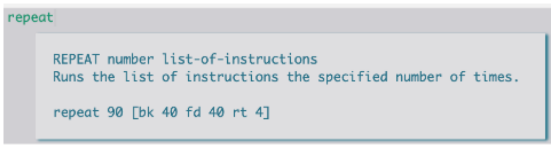
Instead of copying ALL the objects on a Page, you can copy a specific turtle, button - any object you want! It will have all the same features as the original object.

Follow the 3 steps above but instead Select the specific turtle, button or other object you want to copy. Next, make sure you have Selected the Page where you want to Paste the copied object.

Yes, you can copy and paste Pages and Objects in a completely different LYNX Project!

List of most common primitives

Remember: you will see them all in Learner Mode Help (click on the 📖 in the bottom-left corner of the LYNX editor), or for a quick description and example, just type the name of the primitive in the Command Center or in the Procedures Pane and leave your mouse pointer over the name of the primitive for a second or two.



```
repeat
REPEAT number list-of-instructions
Runs the list of instructions the specified number of times.

repeat 90 [bk 40 fd 40 rt 4]
```

TURTLE MOVEMENT

forward (fd)	back (bk)	right (rt)	left (lt)	home
glide	setheading (seth)	setpos	setshape (setsh)	pos
setx	sety	xcor	ycor	

TURTLE STATE

st	ht	setsize		

TURTLE DRAWINGS

pendown (pd)	penup (pu)	penerase (pe)	clean	cleargraphics (cg)
setpensize	setcolor (setc)			

TEXT STUFF FOR TEXT BOXES

print (pr)	insert	cleartext (ct)	settext1	

OTHER TEXT STUFF

announce	question	answer	show	say

CONDITIONALS & OTHER STUFF FOR MAKING THINGS HAPPEN

if	ifelse	repeat	wait	forever
launch	stopall			

OTHER STUFF FOR TURTLES

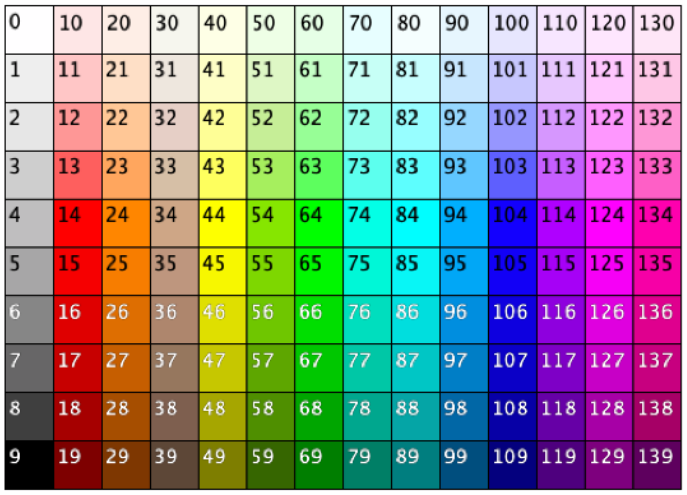
everyone	ask	talkto (tto)	clickon	clickoff

OTHER "RANDOM" STUFF

random	pick			

Color Names and Numbers

LYNX turtles and page backgrounds can have many colors. Colors in LYNX are numbered like this:



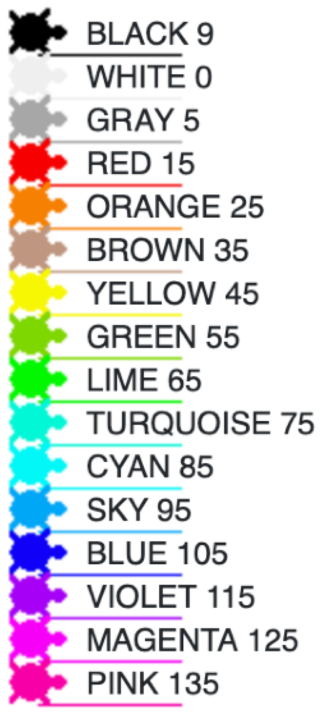
0	10	20	30	40	50	60	70	80	90	100	110	120	130
1	11	21	31	41	51	61	71	81	91	101	111	121	131
2	12	22	32	42	52	62	72	82	92	102	112	122	132
3	13	23	33	43	53	63	73	83	93	103	113	123	133
4	14	24	34	44	54	64	74	84	94	104	114	124	134
5	15	25	35	45	55	65	75	85	95	105	115	125	135
6	16	26	36	46	56	66	76	86	96	106	116	126	136
7	17	27	37	47	57	67	77	87	97	107	117	127	137
8	18	28	38	48	58	68	78	88	98	108	118	128	138
9	19	29	39	49	59	69	79	89	99	109	119	129	139

To set a turtle or background to a particular color, type a `setc` or `setbg` command, followed by the chosen color number:

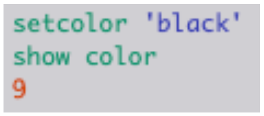
```
setc 127
```

Each set of ten color numbers is for the shades of a particular color. For example, shades of yellow are from 40 to 49 and shades of orange — from 20 to 29. The smaller the number in a set of ten, the lighter the shade and the bigger the number, the darker the shade.

16 colors are considered “basic”. They have not only numbers, but also names. These colors and their names are shown at the left. For these colors you can type `setc color_name` in the Command Center instead of just `setc color_number`



BLACK 9
WHITE 0
GRAY 5
RED 15
ORANGE 25
BROWN 35
YELLOW 45
GREEN 55
LIME 65
TURQUOISE 75
CYAN 85
SKY 95
BLUE 105
VIOLET 115
MAGENTA 125
PINK 135



```
setcolor 'black'
show color
9
```

For example:
`setc 'violet'` and `setc 115`
do the same thing.

If you want your turtle to be one shade darker than the VIOLET color that you see at the left, you need to run
`setc 116`. *No name is available for that shade*.

When to use names and when to use numbers? Names only exist for 16 basic shades. Using names makes your code more readable. But when you do calculations, use numbers, like this:

```
repeat 140 [setc color + 1 stamp fd 10]
```