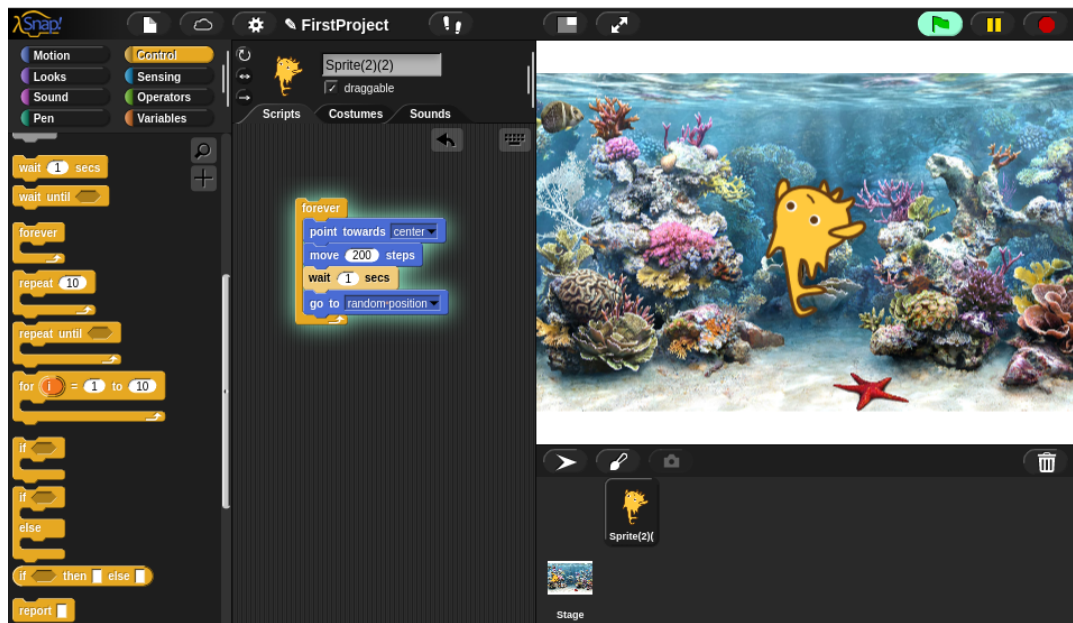


# Snap! Scripting

UBMS Game and Robo Programming with Snap! and Python

June 27, 2023



## Building a script

- To build a **script**, you drag and drop the blocks from the commands area into the script area and **connect** them like a jigsaw puzzle.
- Scripts are **programs**, sets of instructions for the computer. Programs for *interpreted* languages are sometimes called *scripts* (like: R, Python, Snap!, JavaScript).

- Interpreted languages offer instant gratification while compiled languages (C/C++, Java, FORTRAN) need the source code to be translated to machine code, which is then run.

- Python example:

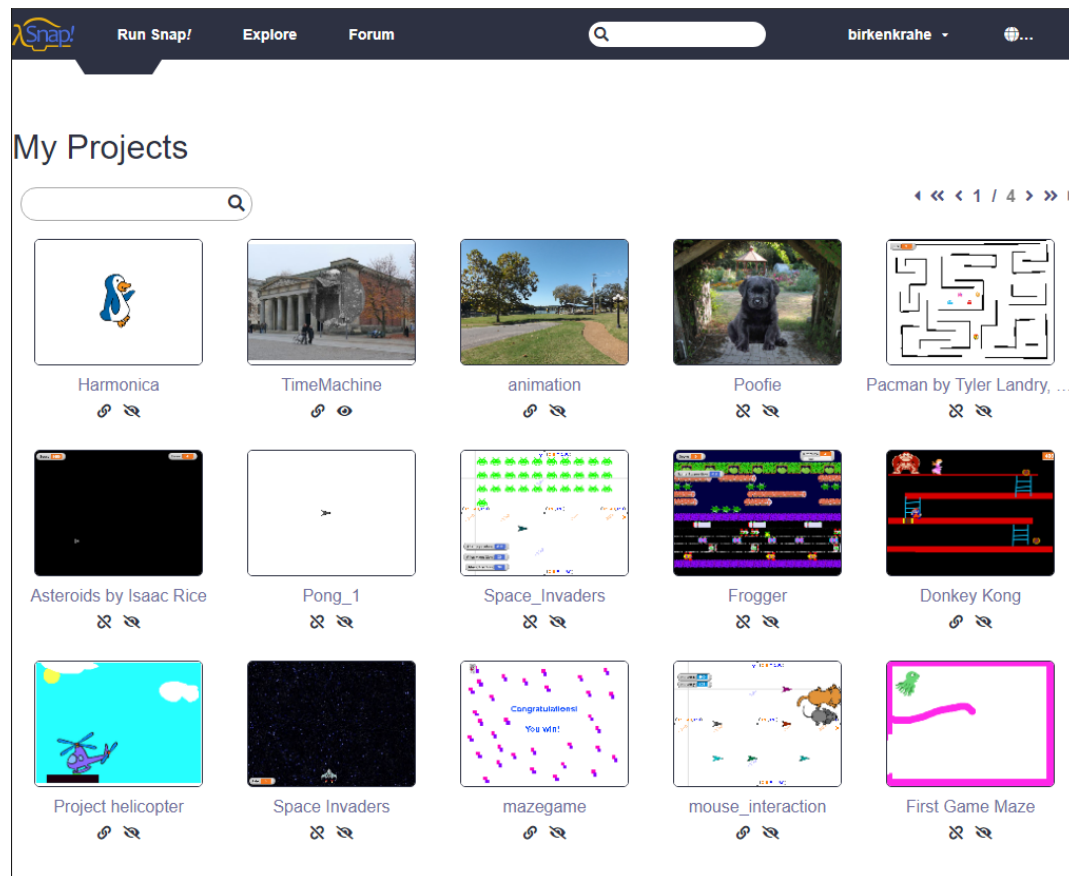
```
print('Hello, world!')
```

- C example:

```
printf("Hello, world!\n");
```

- Programs need to be absolutely **flawless**: you need to be 100% diligent and careful when programming. Otherwise, the computer will refuse to cooperate.
- Modern programming editors (like Emacs here) provide *syntax highlighting* as a visual aid to help satisfy syntax requirements.

## Saving a Snap! project



- Open your Snap! editor now to follow along as I explain things.
- A Snap! *project* is a collection of scripts for sprites.
- You can save your projects in your cloud account (if you are using the cloud version of Snap!), or you can save it locally as an XML file.
- What is XML? (Here is an excellent tutorial overview.)

XML, or eXtensible Markup Language is a layout language that looks a lot like HTML (HyperText Markup Language), but instead of web page display its focus is on wrapping layout information in text-based, tagged files.

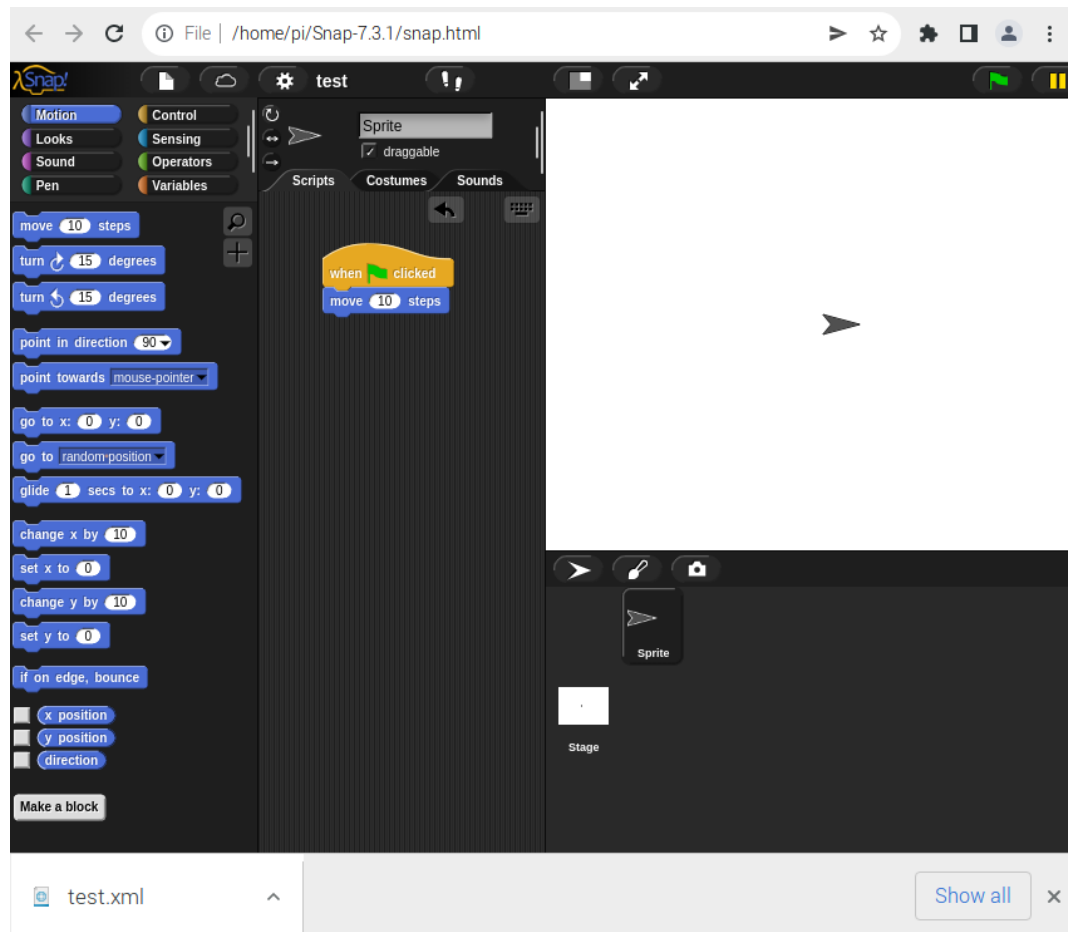
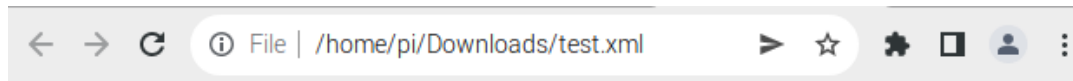


Figure 1: Snap! project example

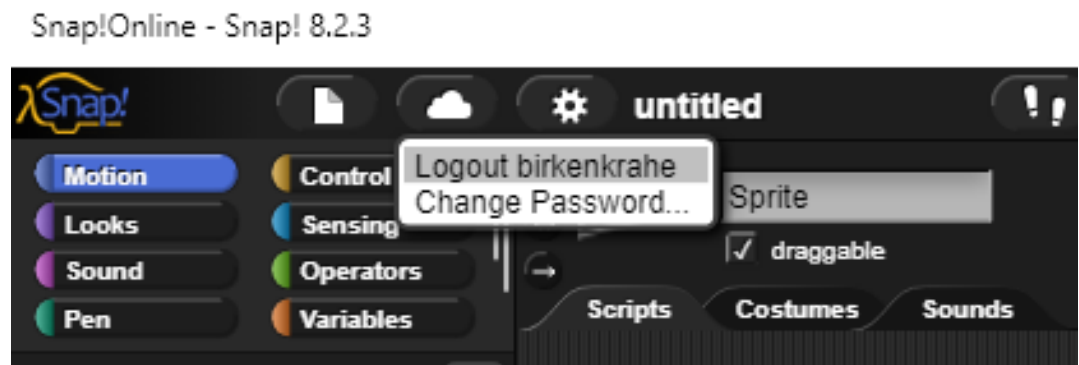


This XML file does not appear to have any style information associated with it. The document tree is shown below.

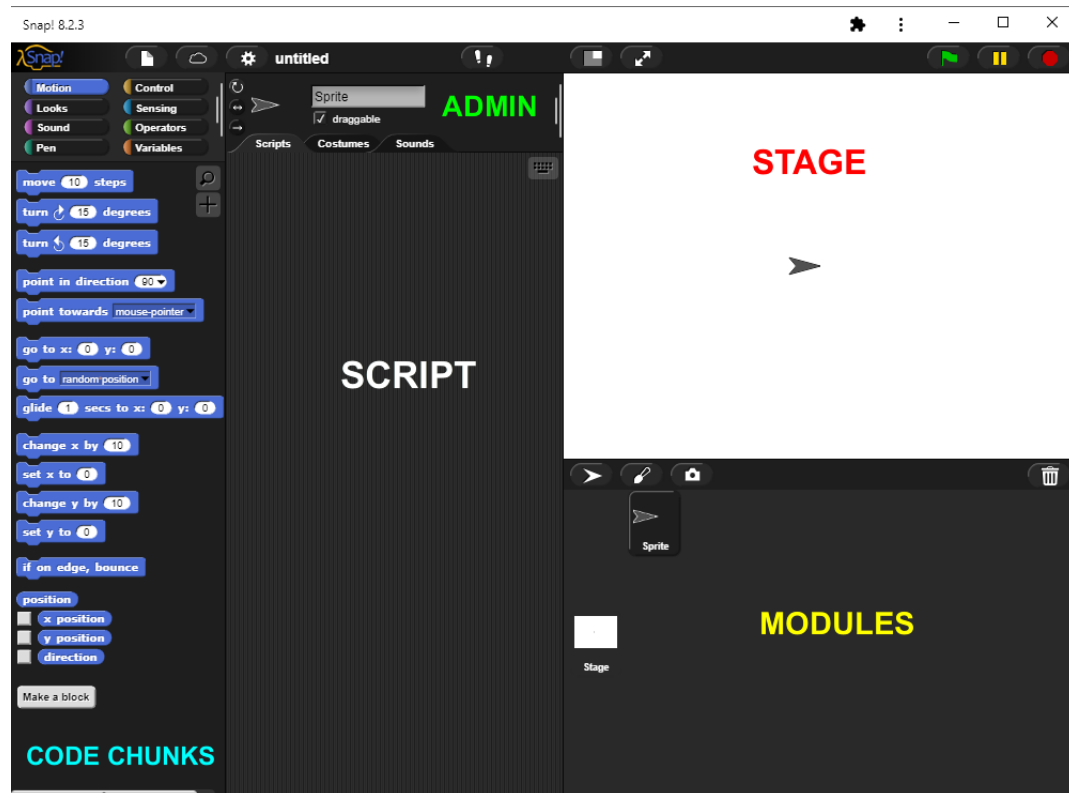
```
▼<project name="test" app="Snap! 7, https://snap.berkeley.edu" version="2">
  <notes/>
  <thumbnail>data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAKAAAAB4CAYAAAB1ov
  </thumbnail>
  ▼<scenes select="1">
    ▼<scene name="test">
      <notes/>
      <hidden/>
      <headers/>
      <code/>
      <blocks/>
      ▼<stage name="Stage" width="480" height="360" costume="0"
        color="255,255,255,1" tempo="60" threadsafe="false" penlog="false"
        volume="100" pan="0" lines="round" ternary="false" hyperops="true"
        codify="false" inheritance="true" sublistIDs="false" id="5">
          <pentrails>data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAeAAAFoCAYAA
          </pentrails>
          ▼<costumes>
            <list struct="atomic" id="6"/>
          </costumes>
          ▼<sounds>
            <list struct="atomic" id="7"/>
          </sounds>
          <variables/>
          <blocks/>
          <scripts/>
          ▼<sprites select="1">
            ▼<sprite name="Sprite" idx="1" x="10" y="0" heading="90" scale="1"
              volume="100" pan="0" rotation="1" draggable="true" costume="0"
              color="80,80,80,1" pen="tip" id="12">
                ▼<costumes>
                  <list struct="atomic" id="13"/>
                </costumes>
                ▼<sounds>
                  <list struct="atomic" id="14"/>
                </sounds>
                <blocks/>
                <variables/>
```

Figure 2: Snap! project example

- To save to the cloud, you need to be registered and logged in. To log in or out, click on the cloud symbol in the editor.



## The Snap! editor



- The **code chunks** contain executable code (lines or clauses)
- The **script** area is where you place the code chunks
- The **stage** is where you see the results
- The **modules** area is where you see all your characters and scenes

## Sprites and costumes



- When you add a new sprite, it always comes up as a "Turtle", a triangular shape.
- Every new Turtle sprite appears at a random place on the screen, facing a random direction, and has a random color.
- Click on the turtle symbol below the stage to create additional sprites or characters.
- You can also use your *camera* to create a sprite.
- To change the appearance of the standard Turtle sprite, load a new costume. There are many ready-made costumes provided.
- This is where the costumes library resides on my computer at home (because I downloaded the Snap! source code): `/home/pi/Snap-7.3.1/Costumes`.
- The computer needs to keep track of all its files. To do this, it uses a hierarchy, like a tree turned upside down, with the *root* at the top. This particular address, `/home/pi/Snap-7.3.1/Costumes` means that the costumes files are located in a directory `/Snap-7.3.1` (which contains all files for the Snap! version 7.3.1), which is contained in a directory



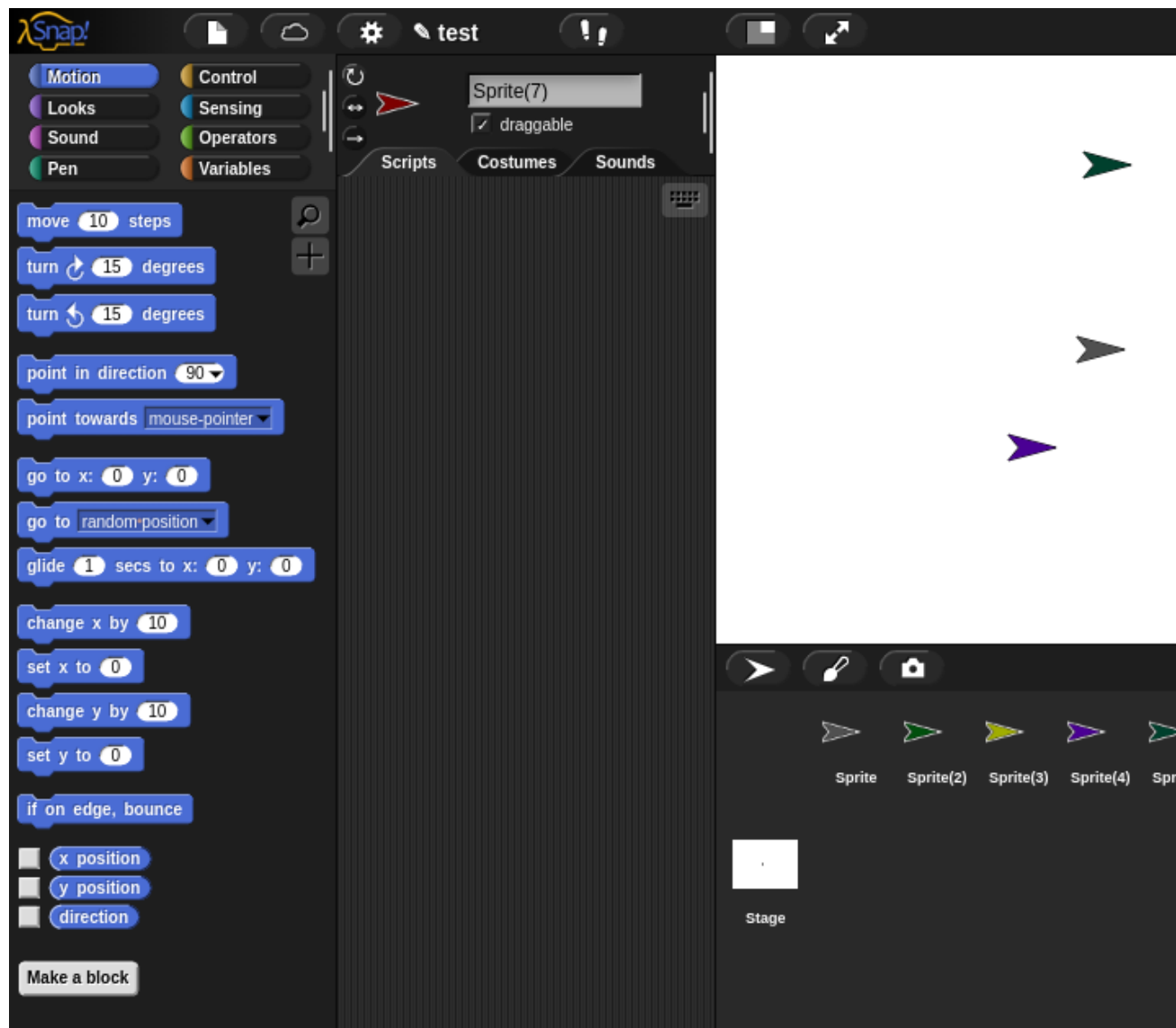


Figure 3: Snap! sprites.

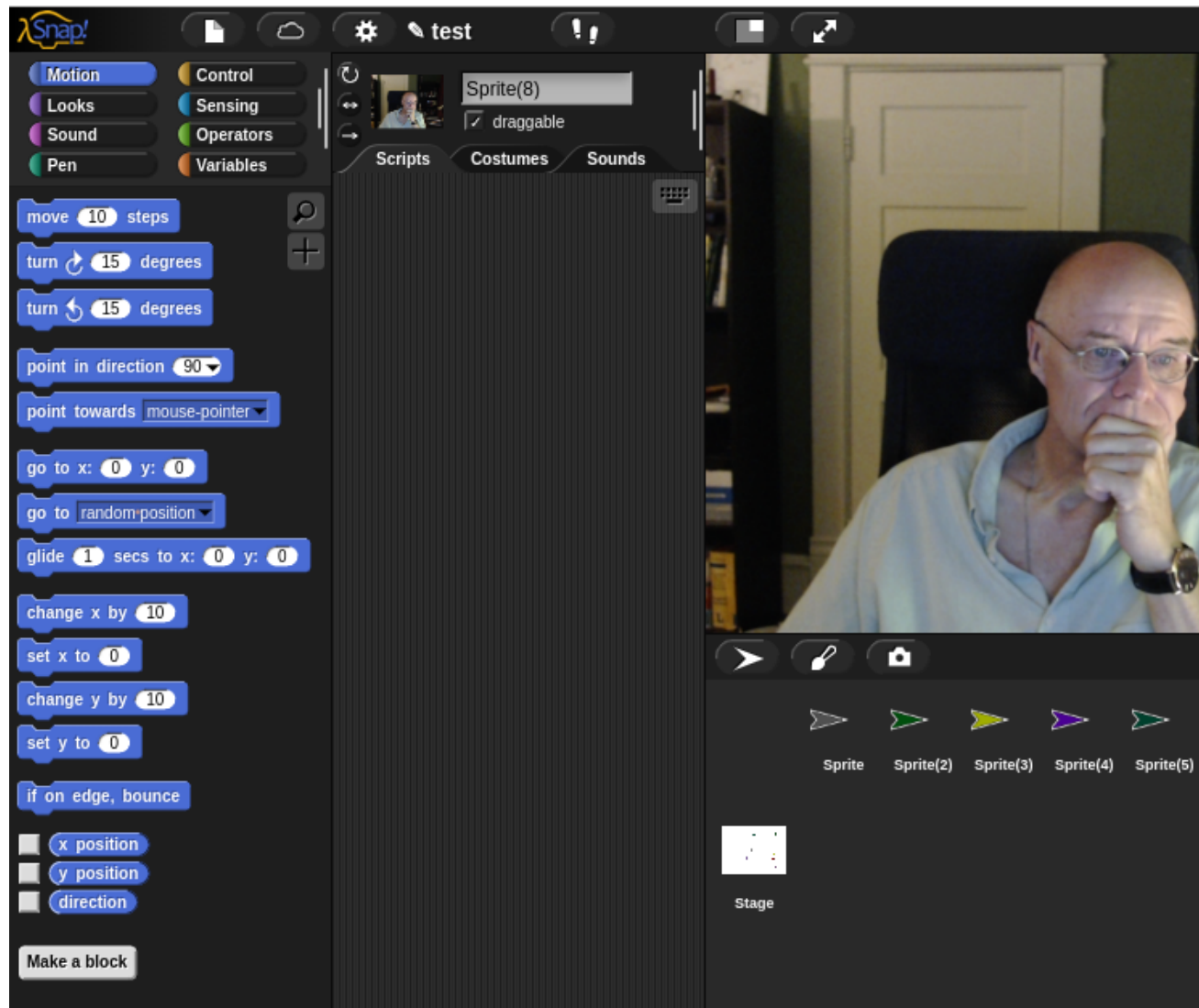


Figure 4: Snap! sprite, created with camera

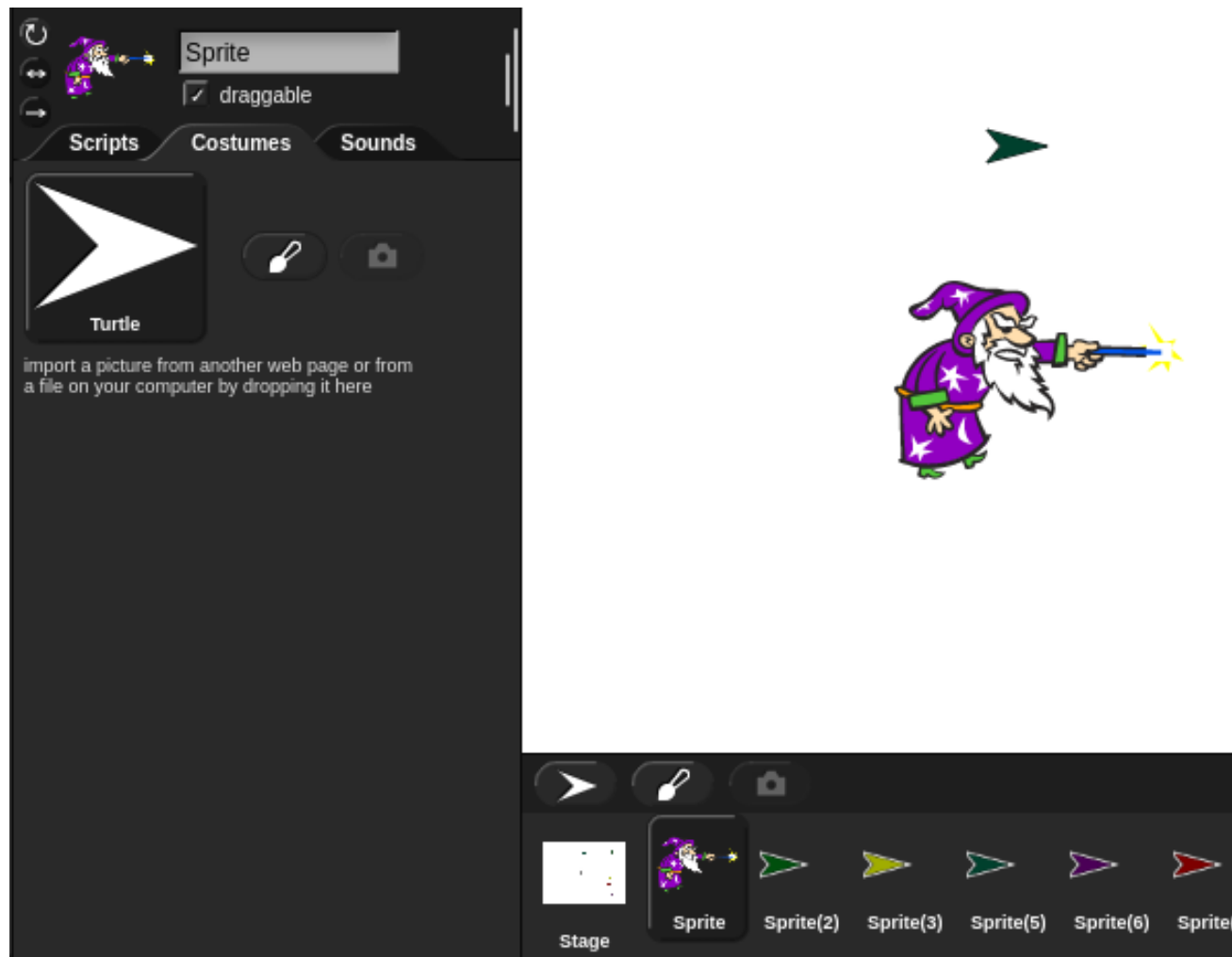


Figure 5: Snap! costume from the media library

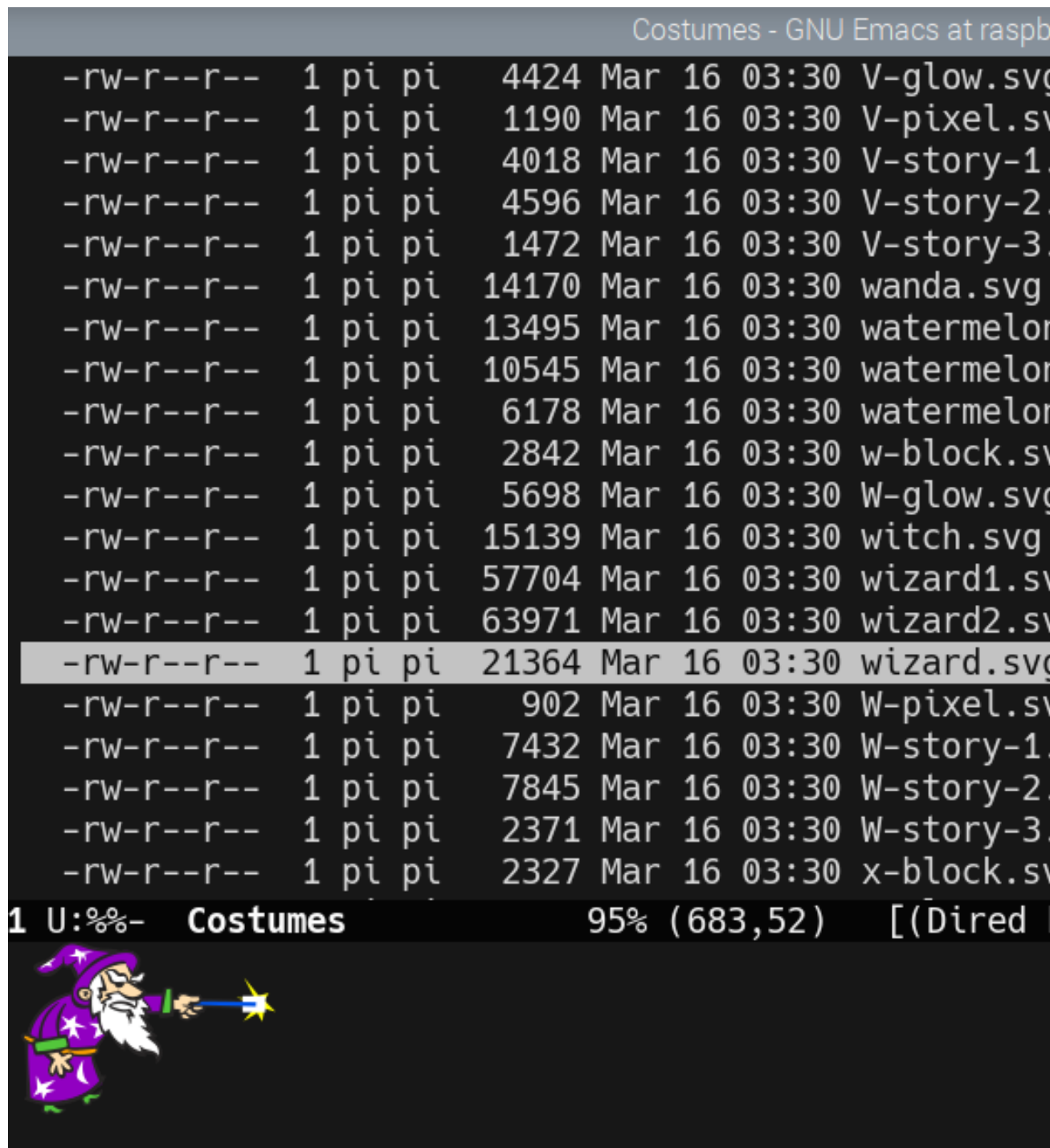


Figure 6: Snap! top menu

/pi (that's my username on this computer), which is contained in the directory /home right below the root directory /.

- Back to costumes! You can also create or modify an existing costume using the paint editor.

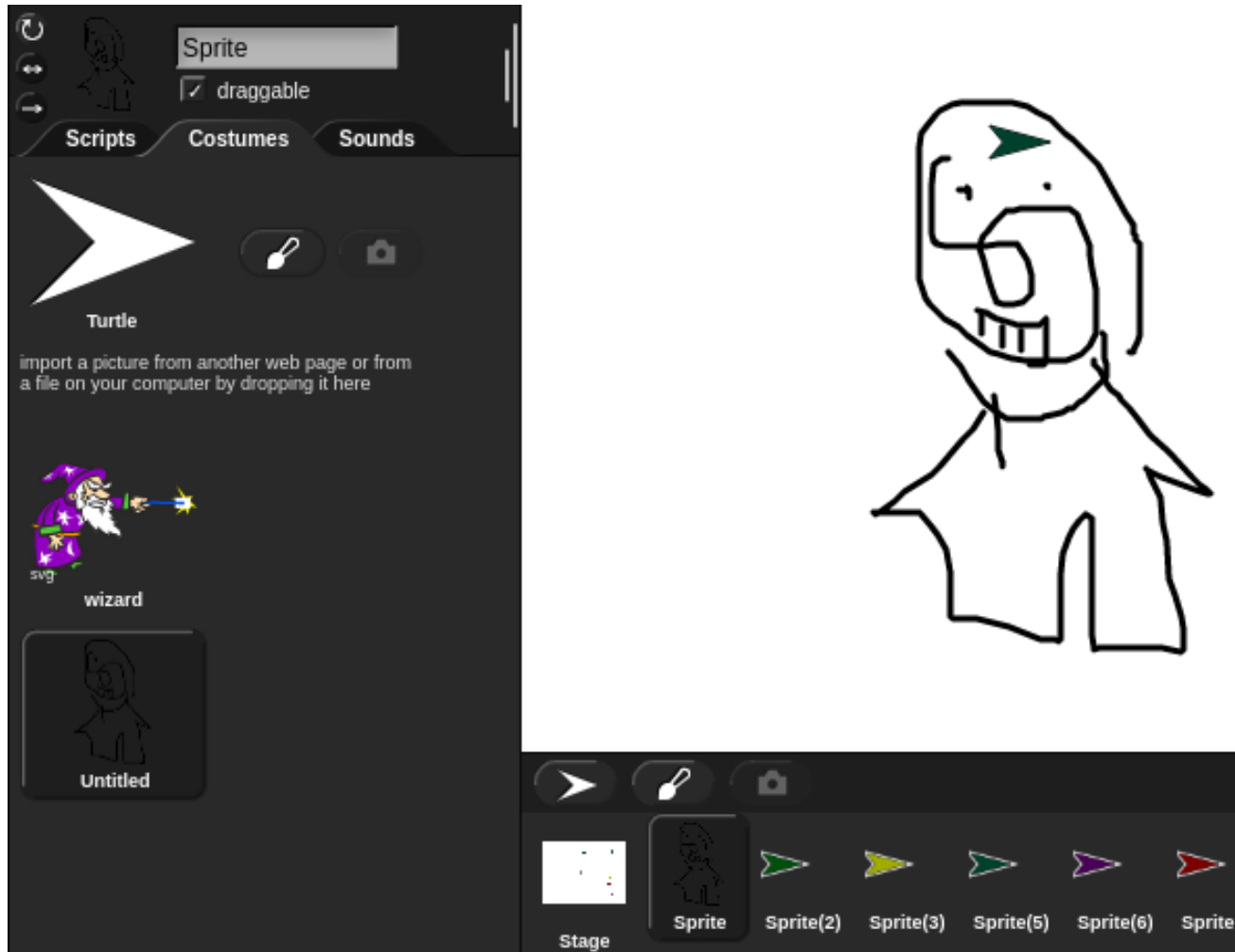


Figure 7: Self-drawn Snap! costume

- To import an image or go to the Costumes library, open the top (or "file") menu next to the Snap! logo, marked by a document symbol.

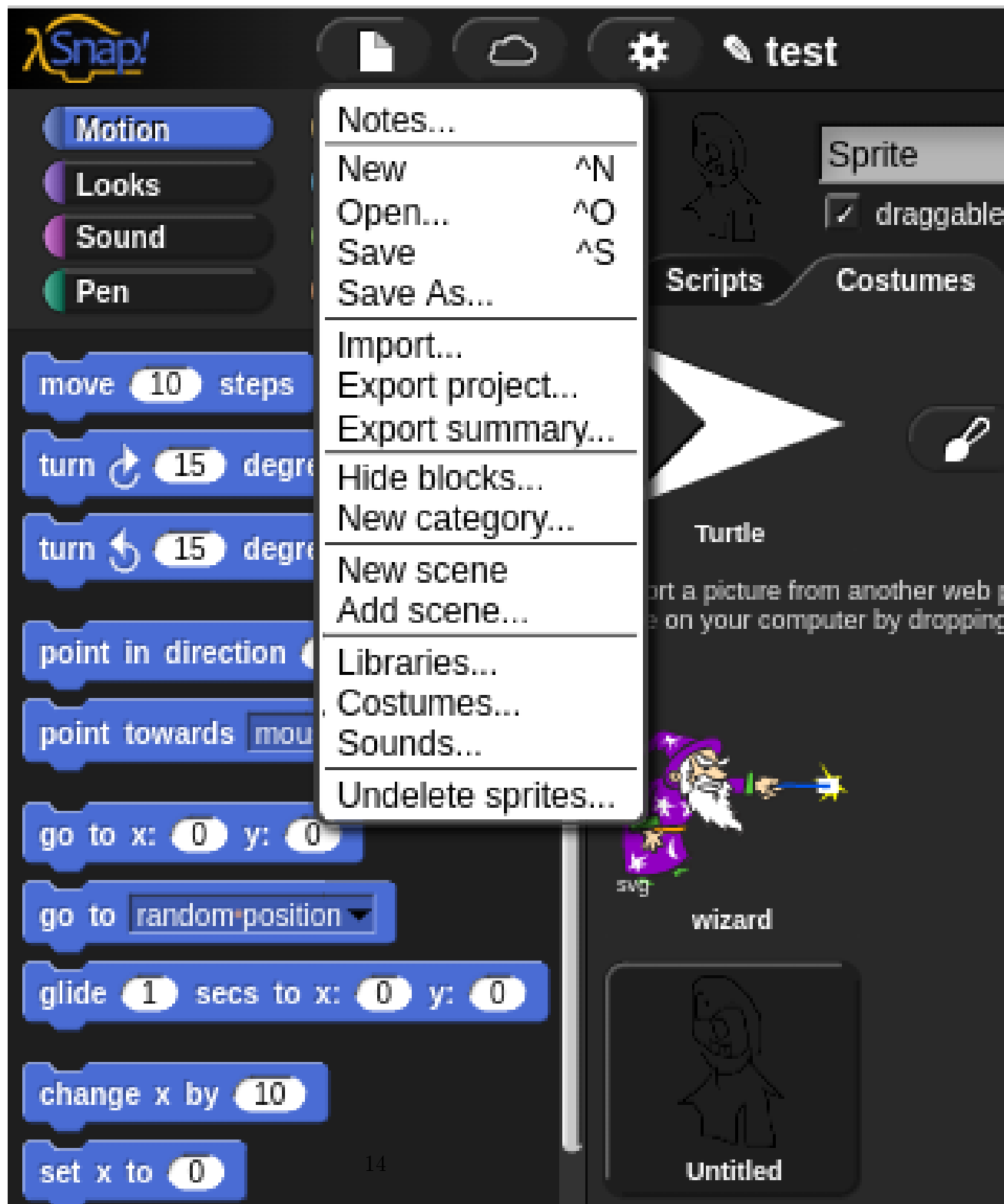


Figure 8: Snap! top menu

## Stage or background

- Similar to the costume library, Snap! comes with backgrounds that you can load for your stage.

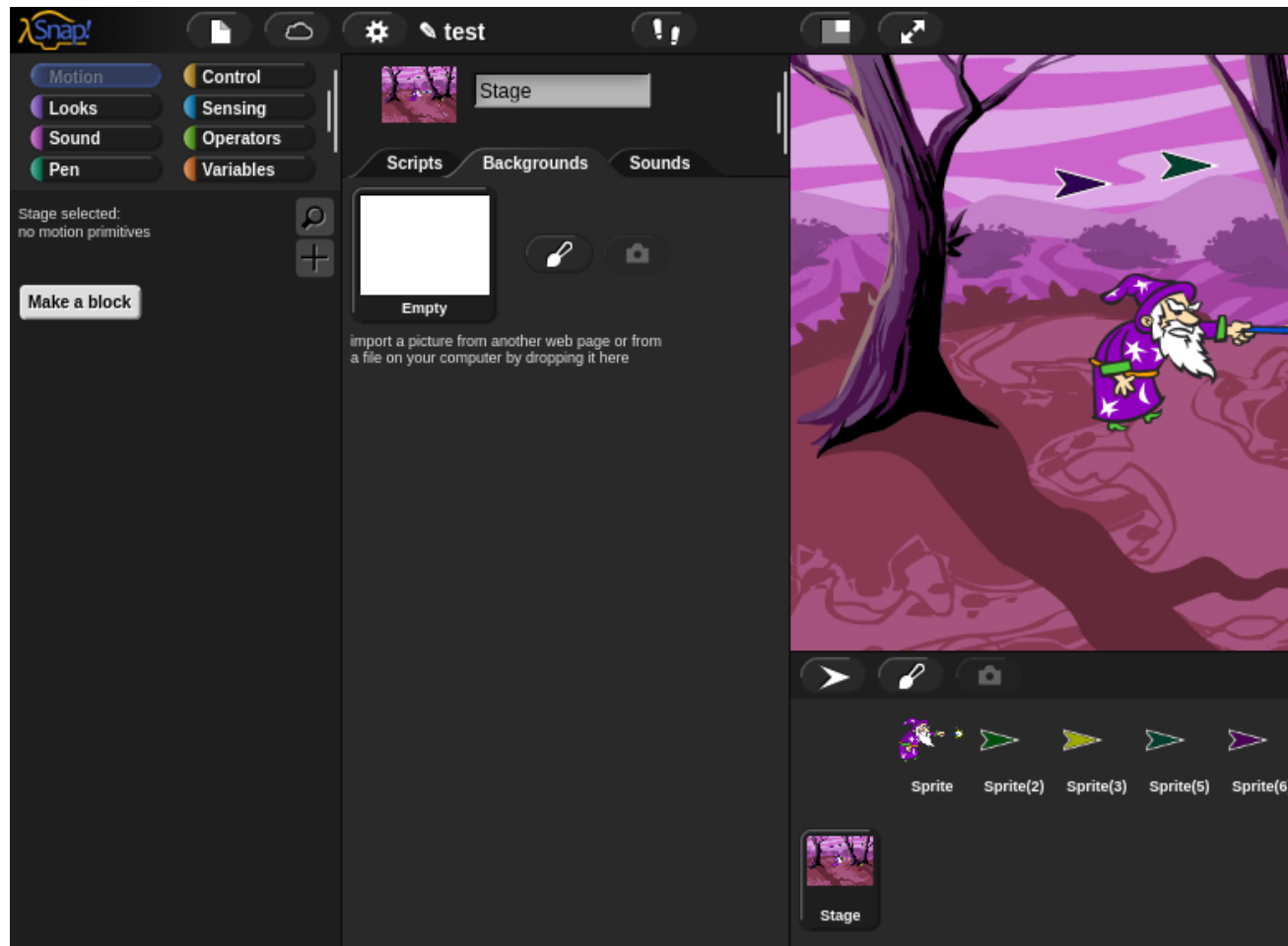


Figure 9: Snap! standard background `woods.gif`

- You can also modify or import backgrounds from your computer.

## Command blocks and scripts

- *Scripts* control the action of sprites (characters).

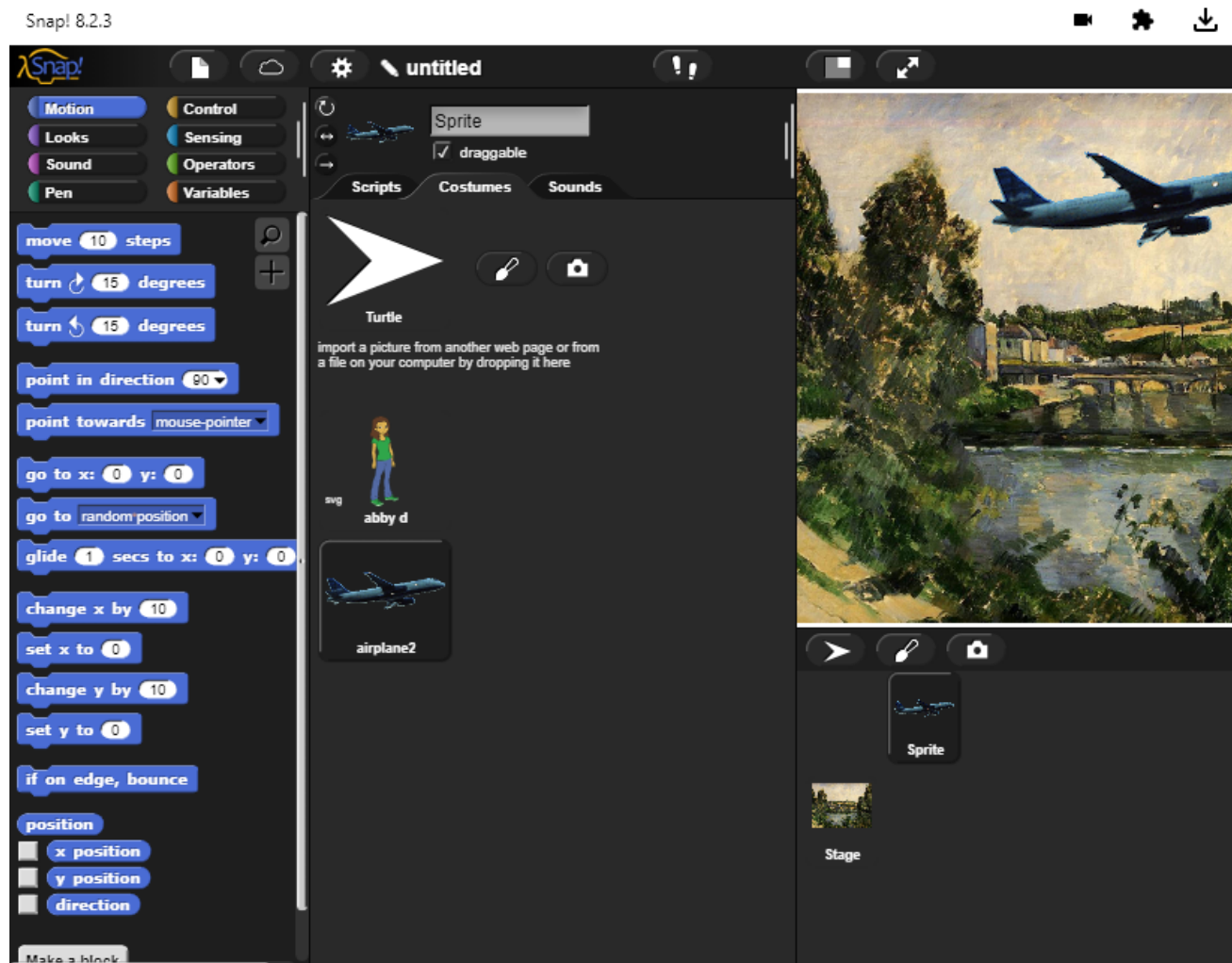


Figure 10: Snap! background changed to a painting by Cezanne.



- *Scripts* are created by dragging command blocks into the script area and snapping them together.
- You can run any command block (aka *programming statement*) by clicking on it. This Gif shows that for "turn 90 degrees".



Figure 11: Snap! motion command to turn sprite clockwise by 90 degrees

`../img/snap_turn.gif`

- When a script is running, the command blocks used are glowing. Clicking on a running script again will stop it.



Figure 12: Snap! motion command that runs forever

## Summary

- Some programming languages (like Snap!, Python) are interpreted, others are compiled.
- Interpreted languages are executed directly, command by command or line by line, while compiled languages produce machine code that is executed (the latter are much faster).
- You can save Snap! "projects" (= collection of programs/scripts) in the cloud or on your local PC.
- XML (eXtensible Markup Language) is a layout language similar to HTML (Hyper Text Markup Language).

- The Snap! editor has 5 different areas: code chunks, script, stage, modules, and admin. Script and stage have costume/background/sound tabs.
- You can create, upload or download sprite costumes, backgrounds and sounds for your project.
- The computer uses a file tree starting at root (its home) to organize content and find files.

## Practice - first script

1. Register an account with `snap.berkeley.edu`. Use your Lyon College email address and FirstnameLastname as Username, e.g. Marcus-Birkenkrahe.

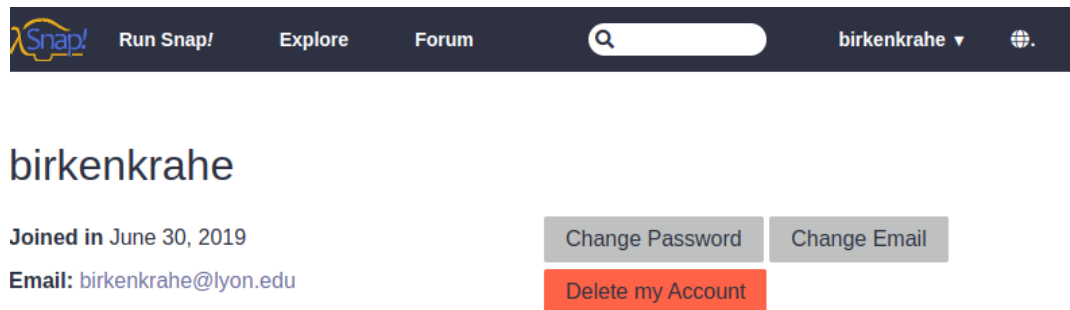


Figure 13: snap.berkeley.edu profile page

2. Create a new named project:
  - Open the main menu at the top
  - Click on **New** (a new project page opens)
  - Click on **Save As ...** and enter the name `FirstProject`
  - Save the project on your computer.
  - Open the file location to see where `FirstProject.xml` was saved
3. Create a new sprite and stage:
  - Add a new *Turtle* sprite

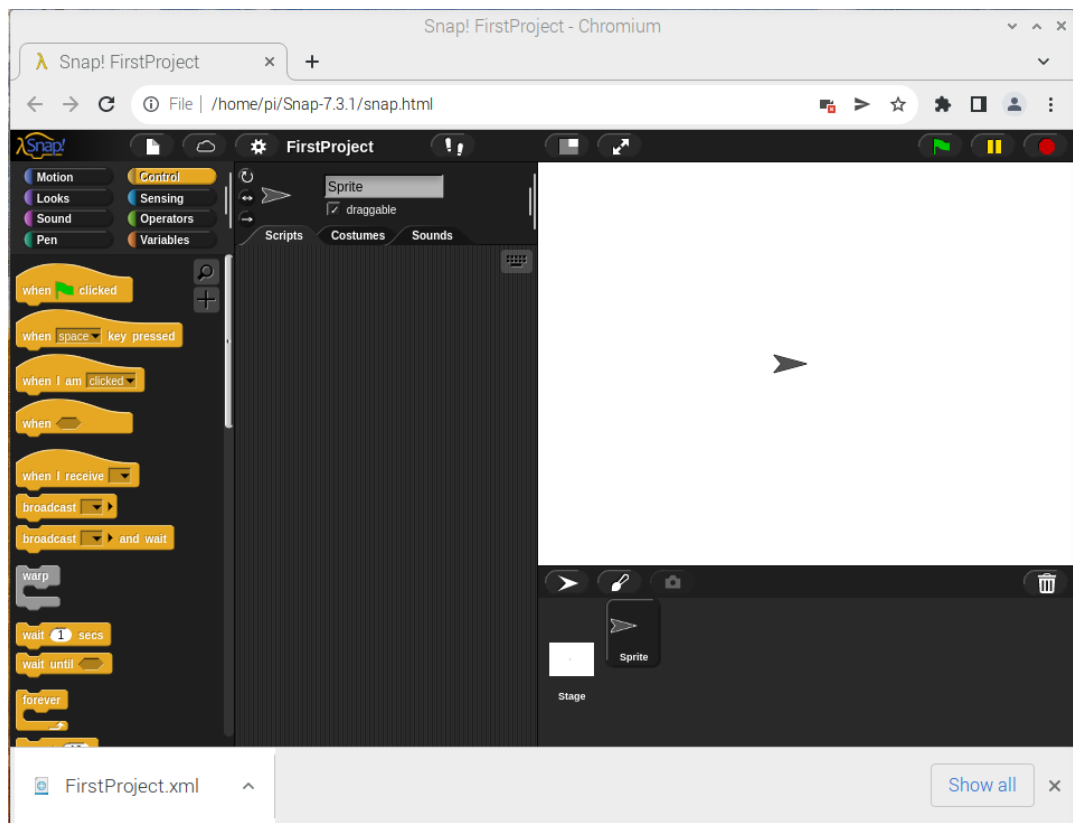


Figure 14: FirstProject in the Snap! desktop app

- Open the *Costumes* menu from the main menu (at the top)
- Click on the sprite icon and pick an animal or human *costume* for the *sprite* using the Costumes library
- Click on the *stage* icon and pick a background for the *stage* using the Backgrounds library
- Save your project to the cloud using **Save As ...** and then choosing the location **Cloud** instead of **Computer**
- Go to **My Projects** on the Snap! website and find your project

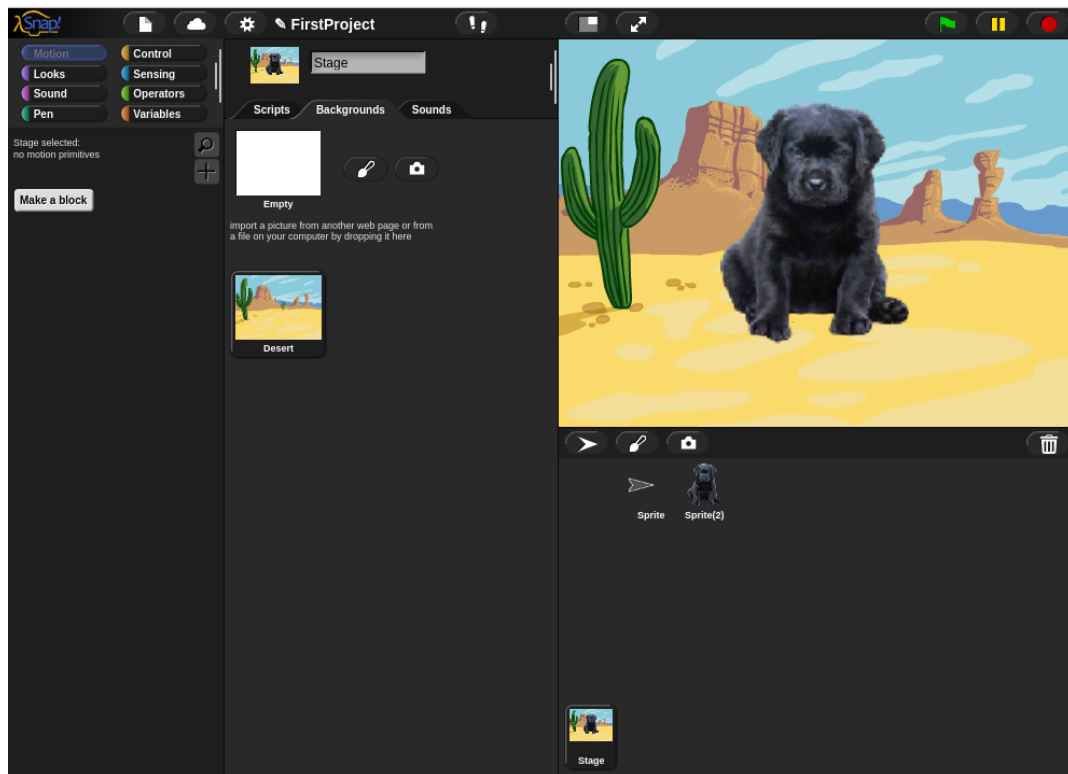


Figure 15: New sprite with new background.

4. Create a simple script with a standard Turtle:
  - Go back to the *Scripts* tab. If the **Motion** command palette is greyed out, then your chosen sprite costume cannot be moved and you need to pick another.

- Make your sprite **point towards center** of the stage
- Make your sprite **move 200 steps**
- Make your sprite **go to a random position**
- Make sure that all your statements/commands are attached to one another in the prescribed order

5. Run script:

- Run the script a few times by clicking on any of the statements in the script
- Go to the **Control** command palette
- Make your sprite **wait 1 secs** between moving and going to a random position
- Run the altered script a few times to make sure it does what it should
- Execute the script **forever** by including it in a **forever** loop
- Stop the program by clicking on the script, or by clicking on the red STOP symbol at the top above the stage
- When running, the final result should look like shown in this video (with your choice of sprite and background, of course)
- Save your project to the cloud location (with **Save As ...**)

6. Share your project and upload the location

- Go to your projects and share the project using the **Share** button.
- You can now publish the project, which means that it will be visible (and searchable) in the Snap! website
- On the project page, you can **Unshare** and **Unpublish** your project.
- On the **My Projects** page, you also see if a project is shared and/or published.
- You can add projects to collections.
- Published projects and collections are displayed on your public page.

# FirstProject



*This project has no notes*

**Created:** July 8, 2022

**Last updated:** July 8, 2022

**Shared:** July 8, 2022

Unshare

Publish

Figure 16: You can share/unshare<sup>22</sup>, and publish/unpublish projects

# My Projects

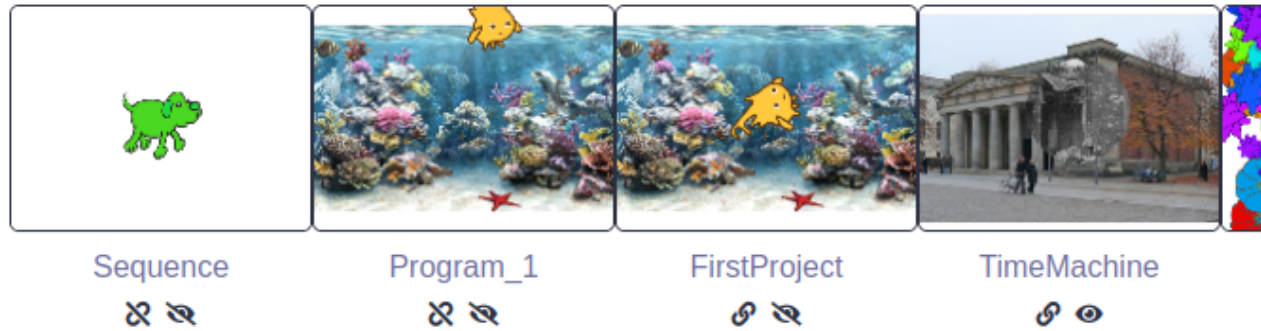
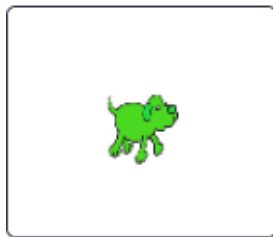


Figure 17: My "My Projects" page

## Practice solution - first script

- Screenshot:

## COR100\_fall2022 by birkenkrahe



Created: July 10, 2022

Last updated: July 10, 2022

Projects for COR 100 Snap!  
Programming Playground class, fall  
2022, Lyon College, Batesville, AR.

birkenkrahe +

Share

Delete



FirstProject



by birkenkrahe



Program\_1




by birkenkrahe

Figure 18: My collection of projects for this course




← → ↻ snap.berkeley.edu/user?user=birkenkrahe

 Run Snap! Explore Forum


## birkenkrahe's public page

### Public Projects



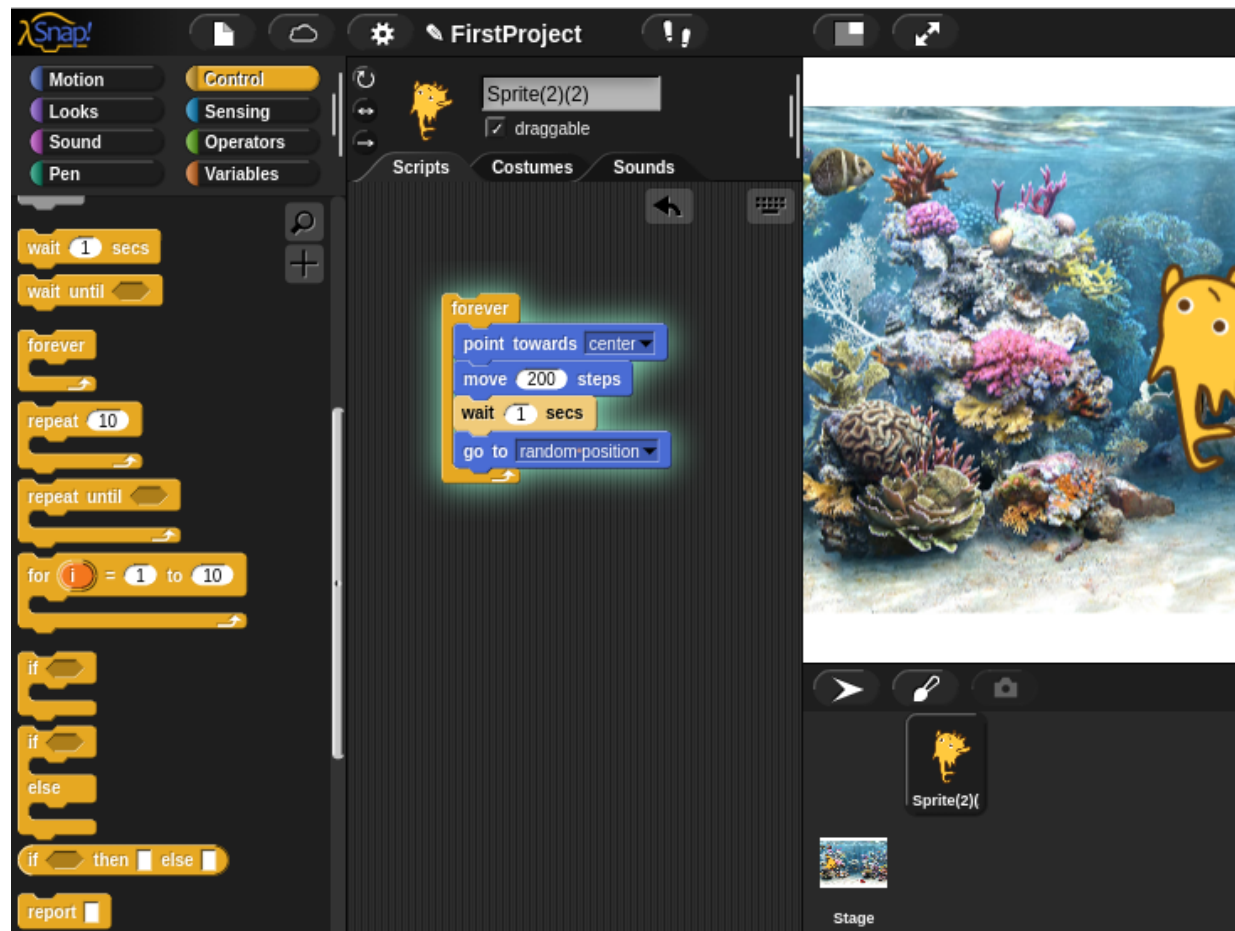
TimeM...

### Public Collections



COR10...  
by  
birkenkrahe

Figure 19: My collection of projects for this course



- YouTube video
- GDrive video
- Project URL