Making Flags

(Also available in Pyret)

Students recreate images of flags of varying complexity by transforming and composing image functions and applying their knowledge of ratios and coordinates to scale and position the shapes precisely.

Lesson Goals	Students will be able to: • Put one image on top of another, using the `put-image` function. • Decompose complex images into parts. • Combine and manipulate images to create more complex images.
Student-facing Goals	• Let's learn how to compose images by placing one image on top of another. • Let's make complex images like flags using code.
Prerequisites	• Simple Data Types • Contracts • Defining Values
Materials	• PDF of all Handouts and Page • Flags Starter File • Flags of Netherland, France & Mauritius Starter File • Lesson Slides • Printable Lesson Plan (a PDF of this web page)

Supplemental Materials	• Additional Printable Pages for Scaffolding and Practice • *Flag of Puerto Rico Starter File* • *Alaskan Flag Starter File* • *Flag of Lebanon Starter File* • *Flag of Mexico Starter File* • *Flag of Trinidad and Tobago Starter File* • *Flag of Turkey Starter File* • *Scaling Flag Ratios (Desmos)* • *Matching Code to Images using overlay and put-image (Desmos)* • *Flag Wizard*
Preparation	• For some students, the use of graph paper for this activity will make a big difference!
Key Points For The Facilitator	• The `put-image` function treats the second image as the first quadrant of a cartesian plane. Having students sketch their flag on a sheet of graph paper drives this point home, and makes the programming easier. • This is an excellent opportunity to introduce students to *indenting code*. The difference between poorly-indented flag code and well-indented flag code is extremely noticable.

Glossary

comments :: messages in the code, generally ignored by the computer, to help people interacting with the code understand what it is doing

define :: to associate a descriptive name with a value

origin :: the point (0,0) where the horizontal and vertical axes intersect

ratio :: the relative sizes of two or more values

scale :: resize an image to be larger or smaller while maintaining ratios and proportions

Putting Images Together *25 minutes*

Overview

Students learn about the `put-image` function.

Launch

As you've already seen, `overlay` sticks two images together, so that the *center* of the first image is placed exactly on top of the *center* of the second image. But what if we want to put the dot somewhere besides the center?

The `put-image` function works like `overlay`, but instead of placing the centers of each image on top of one another, it *translates* the center of the top image by some distance in the x- and y-direction.

Think of the background image as a sheet of graph paper with the ***origin*** (0,0) in the bottom left corner.
The numbers in `put-image` specify a point on that graph paper, with the center of the top image being placed there.

- Take a look at Estimating Coordinates.
- The code beneath each image is missing the x and y coordinates to place the dot.
- Estimate the x- and y-coordinate of the dot's location for each image to complete the code!

Investigate

Open the Flags Starter File and click "Run".

The programmer who wrote this code included a series of special lines in this file - called ***comments*** - that will never be read by the computer.

Professional programmers use comments *all the time*.

Professional programmers work with teams who need to be able to follow each other's thinking in order to collaborate efficiently. Comments are a way for programmers to leave notes for one another, and even for a single programmer to keep track of their thinking when *they* come back to their code another day.

- Each language has its own symbol for commenting. In WeScheme, we use the semi colon (`;`) to tell the computer to ignore what we're typing. What color does the code turn when it has a semicolon in front of it?
  - *Purple!*
- Type `japan-flag` into the Interactions Area. What do you get back?
  - *We see Japan's flag - a white background with a red dot in the center.*
- Type `japan` into the Interactions Area and compare the image to `japan-flag` .
  - *We see a variant of Japan's flag; the dot is in the lower left now.*
- `japan` is composed using `dot` and `background` . Type each of those variables into the Interactions Area. What do you get back?
- Take a look at the `japan` code and fix it so that it matches the `japan-flag` image.
- What is the Contract for `put-image` ?
  - `put-image :: Image, Number, Number, Image`

Next, students will analyze code for the flag of the Netherlands, which uses `put-image` twice.

- Open the Flags of Netherland, France & Mauritius Starter File and click "Run". Look at the code for the flag of the Netherlands. How big is the flag?
  - *The flag is 200 x 300 pixels.*
- What was the programmer thinking when she coded the height of the red stripe as `(/ 200 3)` ?
  - *The red stripe's height is one-third of the total height (200).*
- The center of the blue stripe is placed at (`150` , `(/ 200 6)`). How did the programmer know to use 150 as the x-coordinate?
  - *150 is half of the rectangle's width; that position will horizontally center the blue stripe.*
- What was the programmer thinking when she coded the y-coordinate as `(/ 200 6)` ?
  - *Each stripe occupies 1/3 of the flag's total height. That means the vertical center of the blue stripe is at 1/6 of the flag's height.*
- Explain the thinking behind coding the red stripe's y-coordinate as `(* 5 (/ 200 6))` .
  - *The red strip's vertical center is at 5/6 of the flag's total height.*

Once you've discussed the code for the flag of the Netherlands with your class, have them keep working with this starter file to write code for the flags of Ireland and Mauritius. Some students will make it through the challenges. Some students may only complete one flag. All of them will be synthesizing how to apply put-image and locate images on the coordinate grid.

Synthesize

Could we completely replace `overlay` with `put-image`? Why or why not?

Going Deeper

If you have time, we have lots of additional starter files to push student thinking linked in the additional exercises at the end of this lesson and now would be the time to dive into them!

Making Flags *25 minutes*

Overview

Students focus on decomposing complex images into simple ones and using `put-image` to combine images.

Launch

Let's dig into the process for how the flags we've seen so far were made:

1) Decompose the Image

We observe that the Japanese flag is made up of two simpler images: a black, outline rectangle and a red dot.

2) Define those Parts

We define `dot` and `background`. Once we've defined those images, we test them out in the Interactions Area to make sure they look right!

3) Find the Coordinates

For each image, calculate what the x- and y-coordinates of where the center should be. *TIP: this is a lot easier if you have a sheet of graph paper handy!*

4) Build the Image & Check that it Matches the Target Image

We stack the parts on top of the bottom image using the coordinates we found. *TIP: don't cram all the code into one line! If you break it up into new lines (for example, hitting "Return" before the x-coordinate and after the y-coordinate), you'll notice that the code forms a "staircase" pattern.* Be sure to compare the image you get with the target image!

If you have time, these matching activities will support student thinking.

- Scaling Flag Ratios (Desmos)
- Matching Code to Images using overlay and put-image (Desmos)

Investigate

- Turn to Decomposing Flags, and choose ONE flag to focus on. On the blank lines below, describe the parts that make up that flag.
  - *Note: You may want to project the color version of this page; otherwise, students using black-and-white photocopies will be unable to identify the colors on the the flags.*

- For a more tactile experience, construct your flag using construction paper.
  - *If you have time, this is a valuable activity! The act of physically building the flag from layers of paper makes the layering of the coded images visible and helps students to remember that white space is not just "blank".*
- Once you're done, return to the Flags Starter File and define those parts.
- Then, compose those parts using `put-image`, and make your flag!

The opportunity to focus on a flag of their choosing is a big motivator for students. Coding flags also presents a rare opportunity for students to share a piece of their identity in math class. If you have more time to devote, we highly encourage you to give students the opportunity to create the image of a flag they connect with in some way. Students might choose the flags of countries related to their heritage, a place they've visited, a country they're interested in, or a group they identify with or support.

Encourage students to choose an appropriately challenging flag that will demonstrate what they've learned. The teacher may introduce parameters if necessary, such as "Flags need at least 5 different shapes". Once students have coded their flags, host a tour of the flags of the world in your classroom!

Flags Project

Many teachers enjoy devoting a few days of class to making flags! If you choose to use this task as a summative project, check out the Rubric: Making Flags as a tool to assess student learning.

Note: To reinforce ratio and proportion concepts, have students define the `WIDTH` and `HEIGHT` of their flags as values, and then *replace the numbers in each flag* with expressions relative to width and height. For example, if the `dot` in the Japanese flag is at (150, 100), those numbers would be replaced with `(/ WIDTH 2)` and `(/ HEIGHT 2)`.

Around the World in your Classroom

Be mindful of the fact that not every student will know their family's geographical history, and that immigration can be a sensitive topic for some students. For this reason, it is important that students have the option to choose a flag based on interest instead of just family history.

Be prepared that students might choose flags representing groups other than countries. Indigenous students might choose a flag that represents their indigenous nation or the American Indian Movement. Students might also choose to focus on the pride flag (representing solidarity amongst members of the LGBTQ+ community) or an ethnic flag (representing solidarity amongst members of different ethnic groups, such as the Hispanic flag). There are also some flags that represent political or activist groups. It is important for the teacher to be open to different beliefs and ideologies, but it is also up to the teacher's discretion as to whether or not a flag is appropriate for use in this project. Flags associated with hate groups, or any institution that denies the dignity of other students, are not appropriate.

Optional: Flags of the World and Flag Wizard may be useful to students looking for ideas. Flags are listed with their width:length ***ratios*** in the opposite order of how we define the sides of a flag in code. e.g. 2:3 could be scaled up to a 300 x 200 flag or 8:11 could be scaled up to 330 by 240.

Synthesize

Which flags were the easiest to make? The hardest?

Why is it useful to define each part of the flag first, before stitching the images together?

Additional Exercises

- Flag of Puerto Rico Starter File - This starter file generates a jumbled pile of shapes. Students work to fix the code by resizing, rotating, and correctly locating the components on the background in order to compose an image that looks like the original flag.
- Alaskan Flag Starter File - For a quick dive into why it's more efficient to ***define*** shapes before building the image.
- Flag of Lebanon Starter File - For practice ***scaling*** imported graphics.
- Flag of Mexico Starter File - Similar to the Puerto Rico Flag Starter File, but this one involves an imported image.
- Flag of Trinidad and Tobago Starter File - If you've already studied Pythagorean Theorem and are ready to apply it.
- Flag of Turkey Starter File - For a fun function-composition puzzle.