MasterPiets (A Piet IDE)

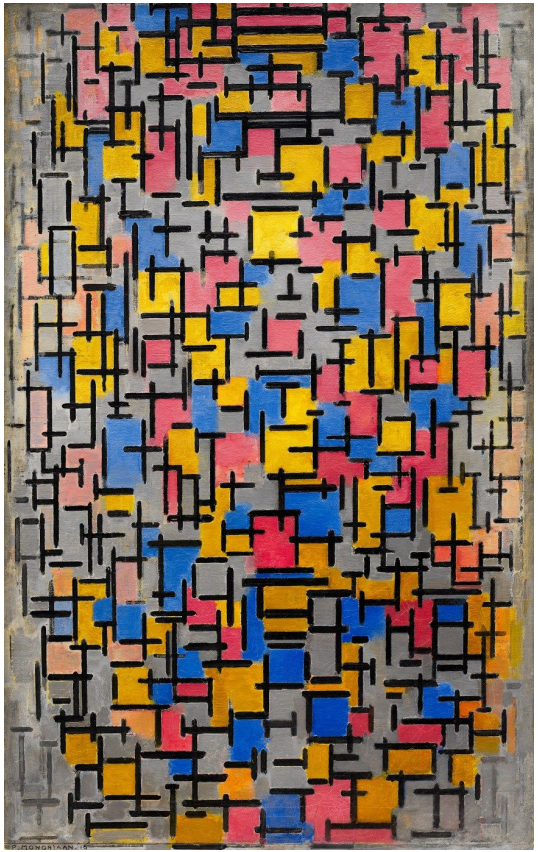
Gabrielle Singh Cadieux

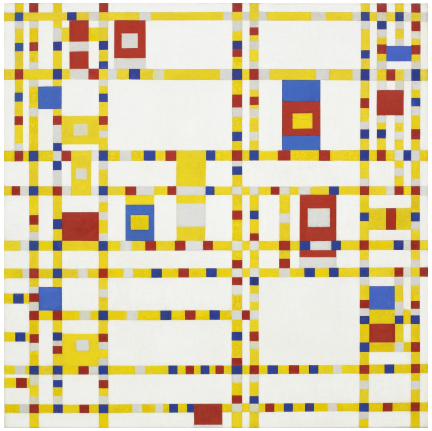
Toronto Hack && Tell: Round 16

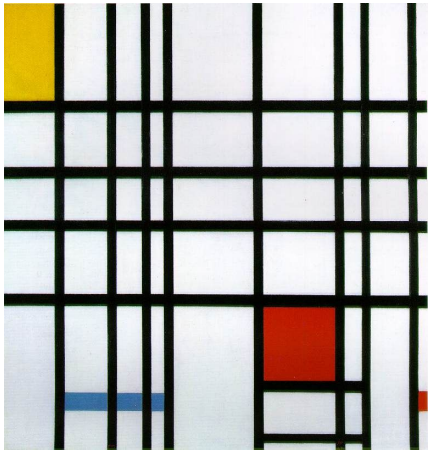
March 6th, 2018

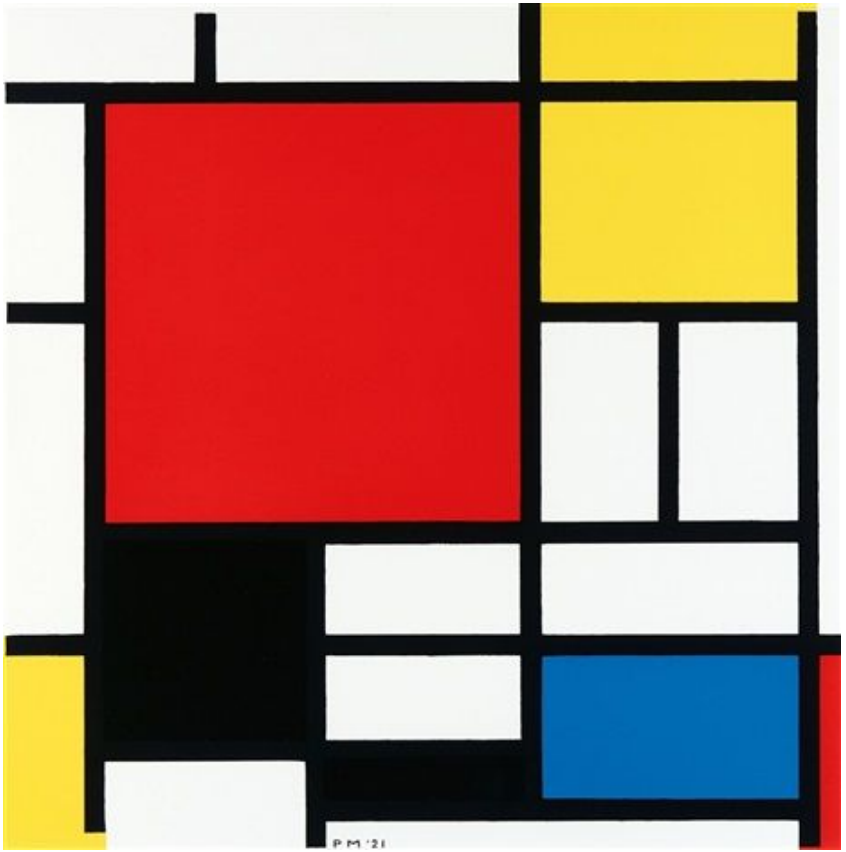
Piet is an esoteric visual programming language

visual: **Piet programs are images**







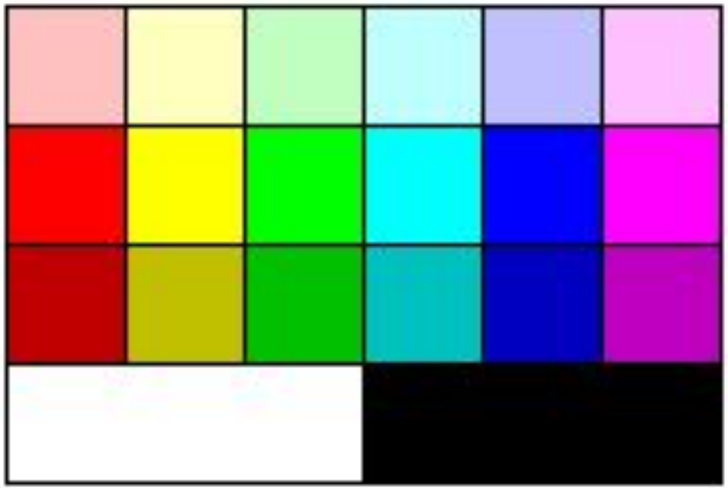


Example works by Mondrian

Piet is named after abstract artist Piet Mondrian

Piet programs contain only these colours:

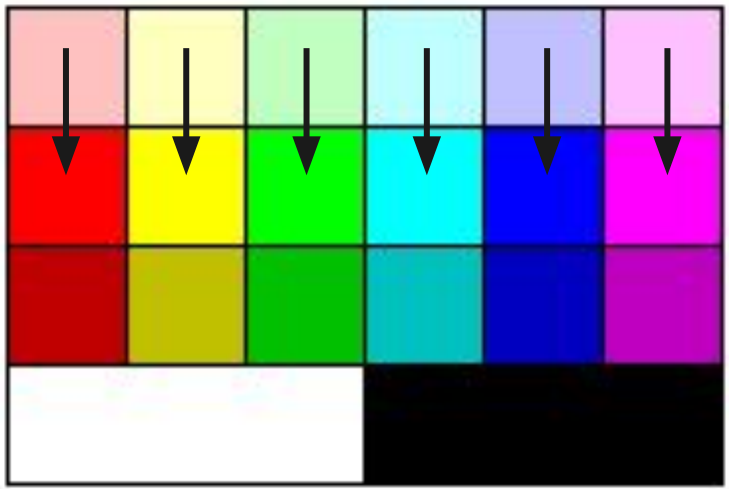
(6 hues x 3 lightness levels + black & white)



Piet palette

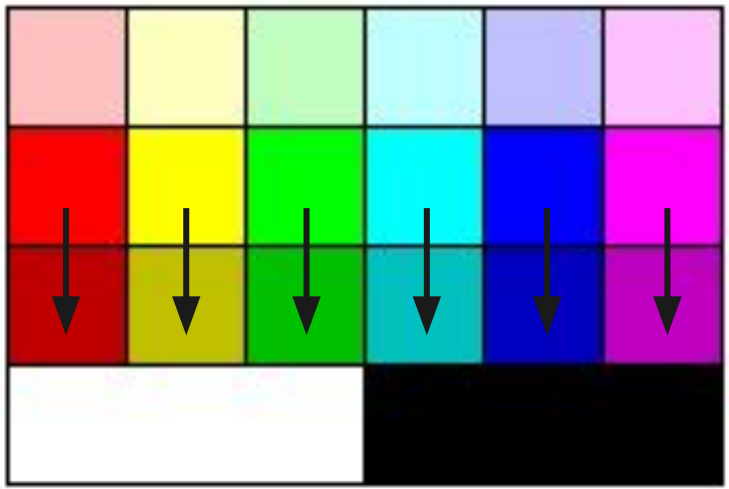
		Lightness change		
		None	**1 darker**	**2 darker**
Hue change	**None**		PUSH	POP
	1 step	ADD	SUB	MUL
	2 steps	DIV	MOD	NOT
	3 steps	>	POINTER	SWITCH
	4 steps	DUP	ROLL	IN(num)
	5 steps	IN(char)	OUT(num)	OUT(char)

Piet instruction set

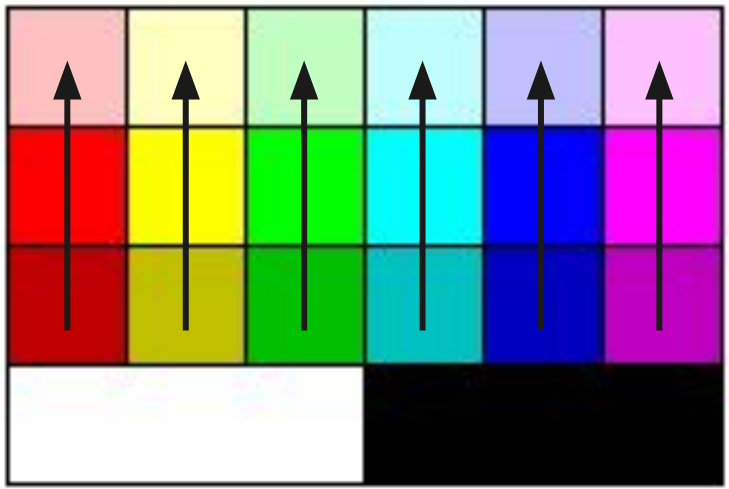


		Lightness change		
		None	**1 darker**	**2 darker**
Hue change	**None**		PUSH	POP
	1 step	ADD	SUB	MUL
	2 steps	DIV	MOD	NOT
	3 steps	>	POINTER	SWITCH
	4 steps	DUP	ROLL	IN(num)
	5 steps	IN(char)	OUT(num)	OUT(char)

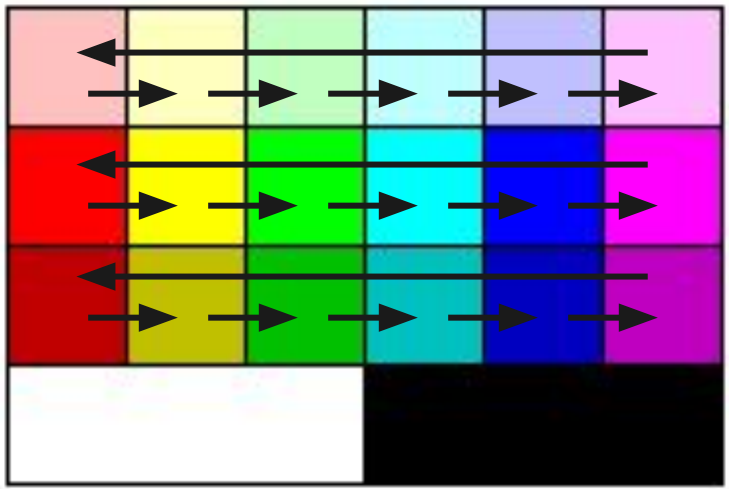
As the Piet interpreter moves around the image,
each **relative transition** between colours produces a command



		Lightness change		
		None	**1 darker**	**2 darker**
Hue change	**None**		PUSH	POP
	1 step	ADD	SUB	MUL
	2 steps	DIV	MOD	NOT
	3 steps	>	POINTER	SWITCH
	4 steps	DUP	ROLL	IN(num)
	5 steps	IN(char)	OUT(num)	OUT(char)



		Lightness change		
		None	**1 darker**	**2 darker**
Hue change	**None**		PUSH	POP
	1 step	ADD	SUB	MUL
	2 steps	DIV	MOD	NOT
	3 steps	>	POINTER	SWITCH
	4 steps	DUP	ROLL	IN(num)
	5 steps	IN(char)	OUT(num)	OUT(char)



		Lightness change		
		None	**1 darker**	**2 darker**
Hue change	**None**		PUSH	POP
	1 step	ADD	SUB	MUL
	2 steps	DIV	MOD	NOT
	3 steps	>	POINTER	SWITCH
	4 steps	DUP	ROLL	IN(num)
	5 steps	IN(char)	OUT(num)	OUT(char)

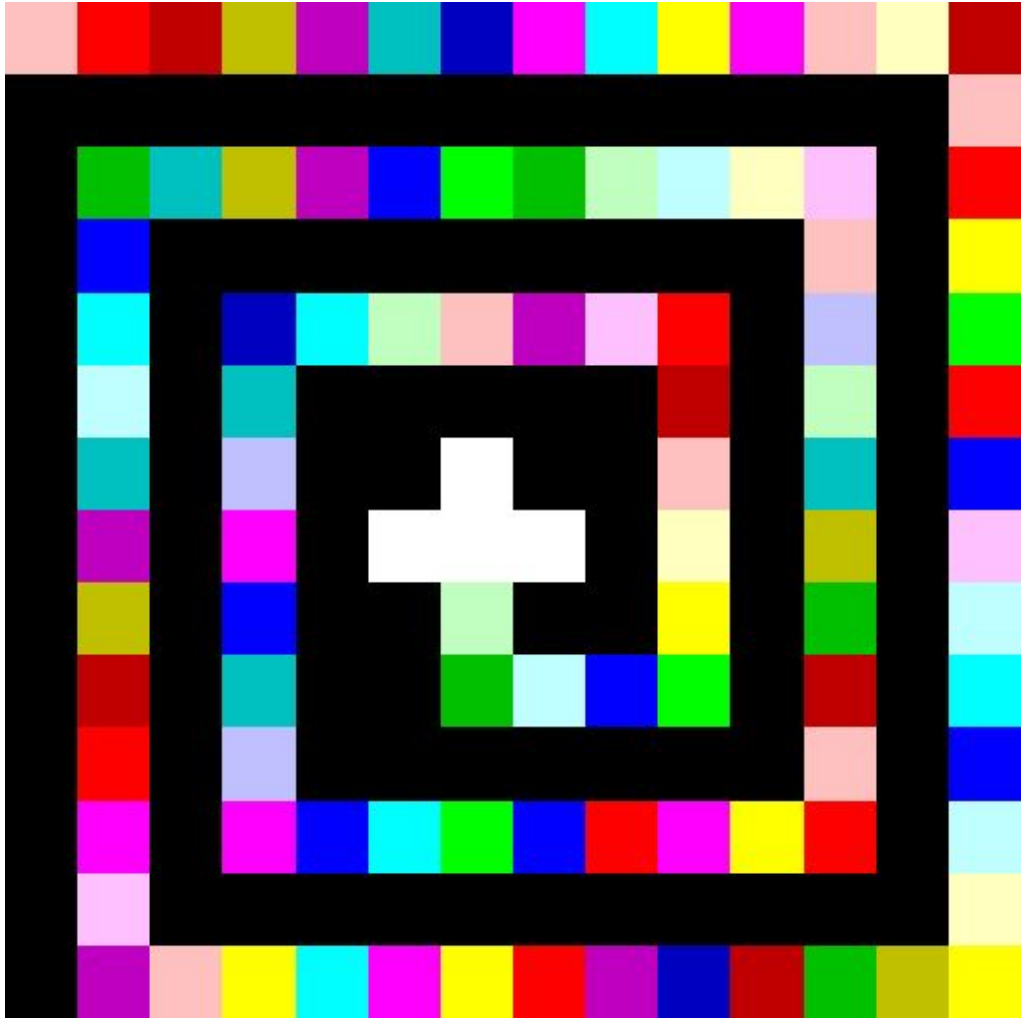
There are so many ways to produce the same command…

Two Piet programs can be functionally-equivalent but look *very* different

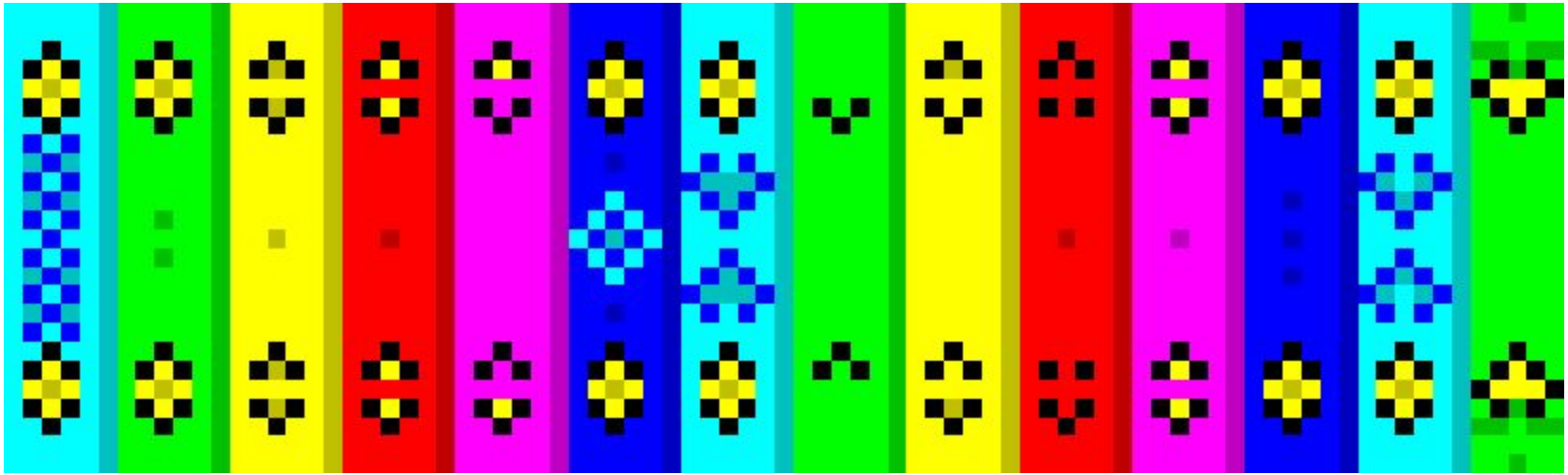
---

Let's try Hello World…

Hello World 1

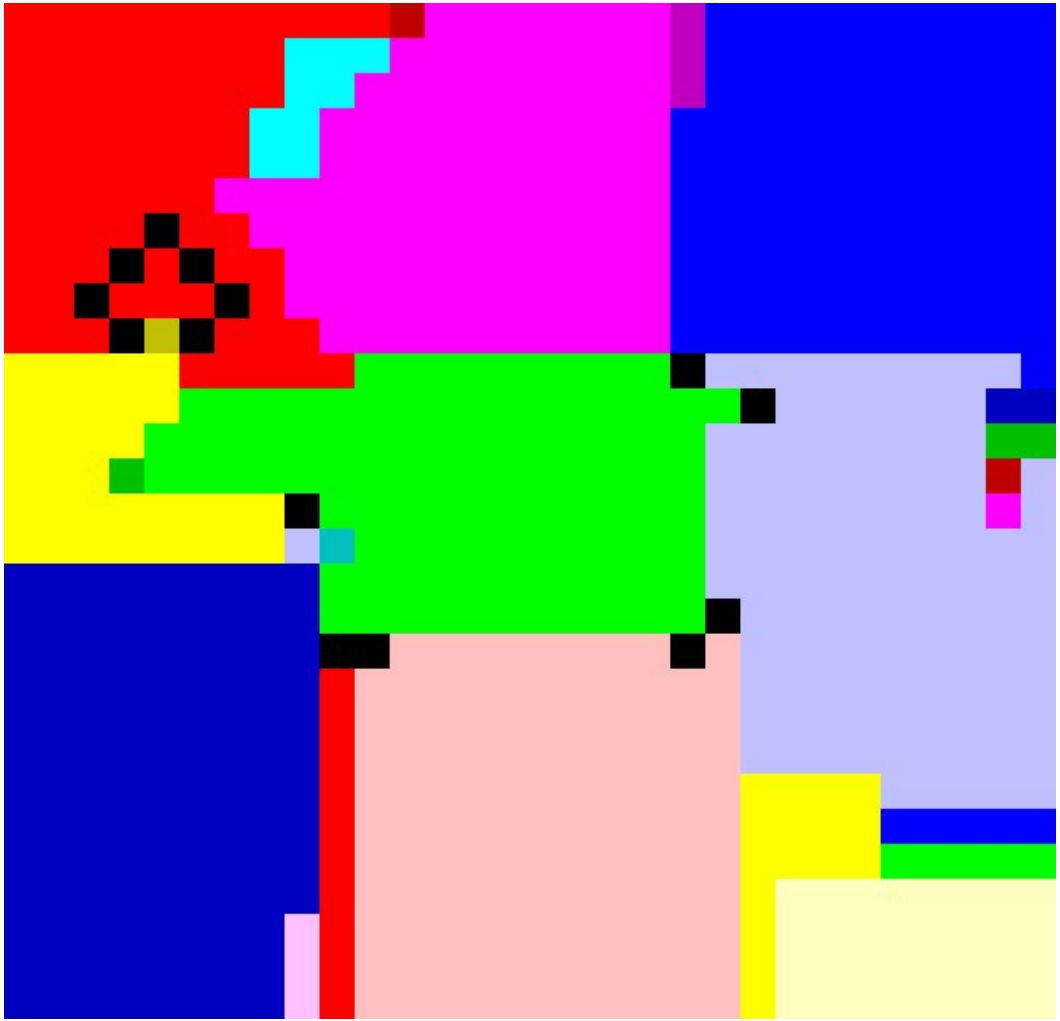


Hello World 2



Hello World 3

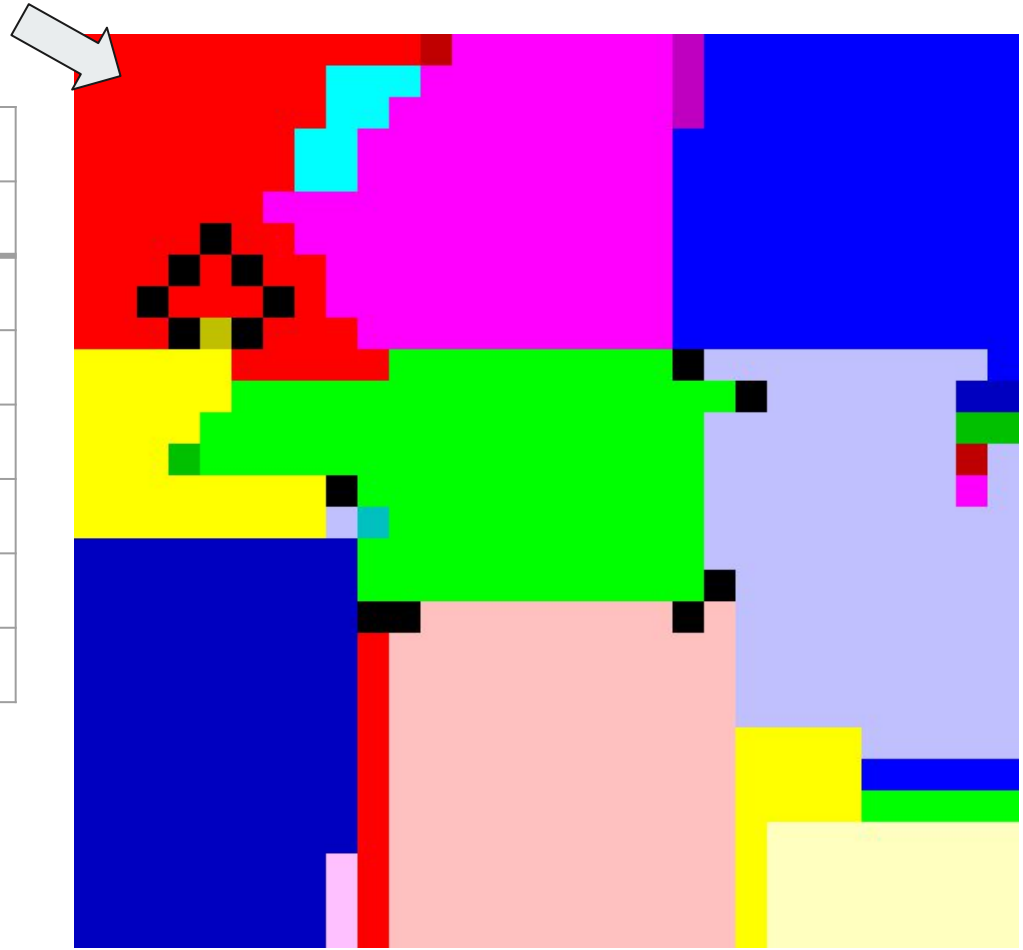
Hello World 4



Hello World 5

Piet interpreter starts at top-left...

		Lightness change		
		None	**1 darker**	**2 darker**
Hue change	**None**		PUSH	POP
	1 step	ADD	SUB	MUL
	2 steps	DIV	MOD	NOT
	3 steps	>	POINTER	SWITCH
	4 steps	DUP	ROLL	IN(num)
	5 steps	IN(char)	OUT(num)	OUT(char)

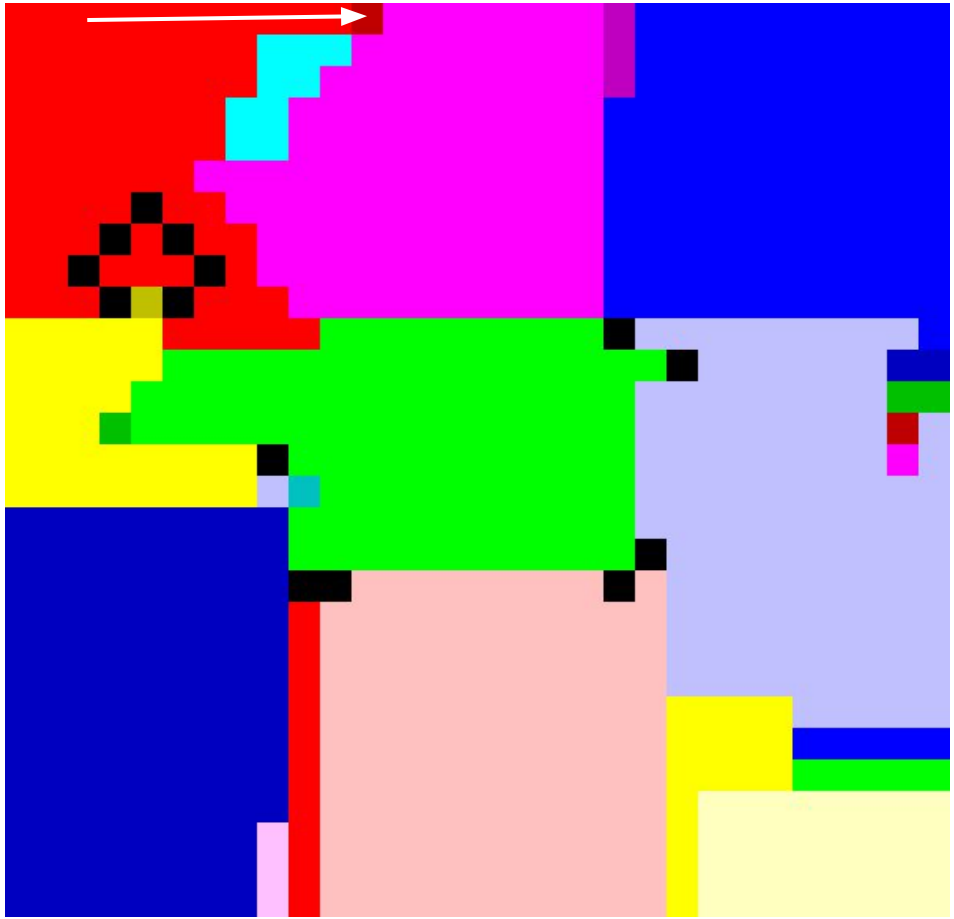


Moves to adjacent block to right...

		Lightness change		
		None	**1 darker**	**2 darker**
Hue change	**None**		PUSH	POP
	1 step	ADD	SUB	MUL
	2 steps	DIV	MOD	NOT
	3 steps	>	POINTER	SWITCH
	4 steps	DUP	ROLL	IN(num)
	5 steps	IN(char)	OUT(num)	OUT(char)

```
PUSH 72
```

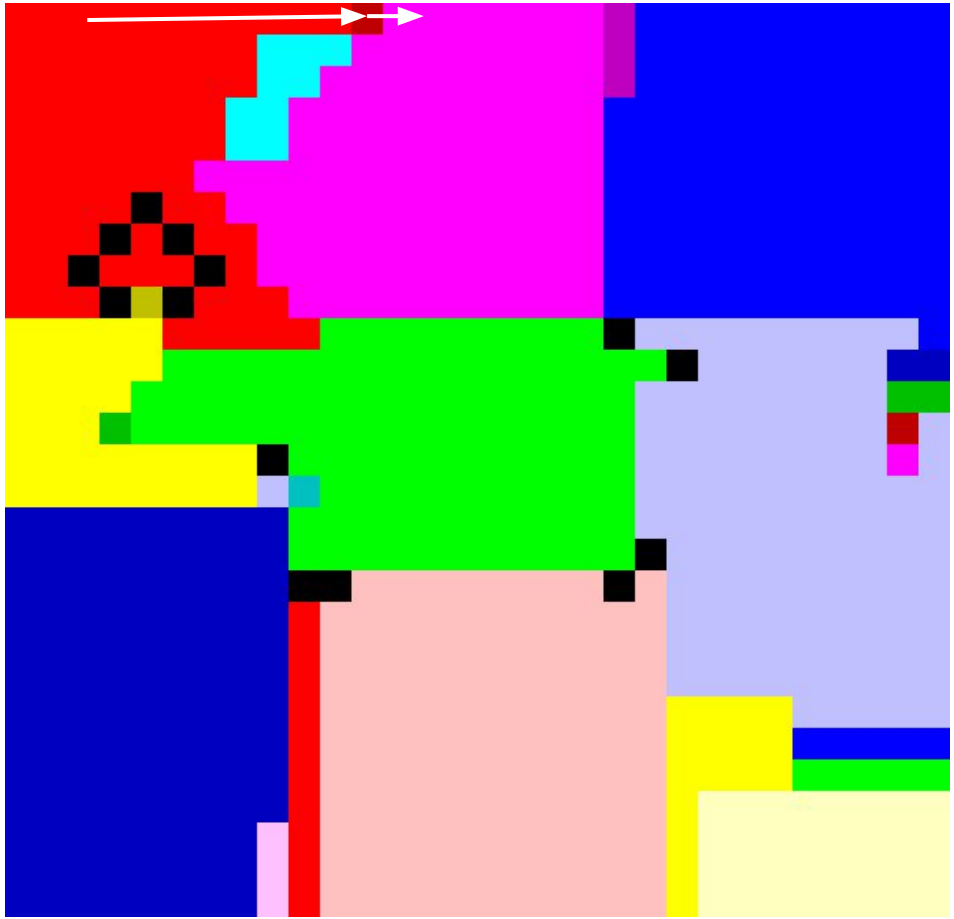
The value pushed is the number of pixels in the block



Continues moving right...

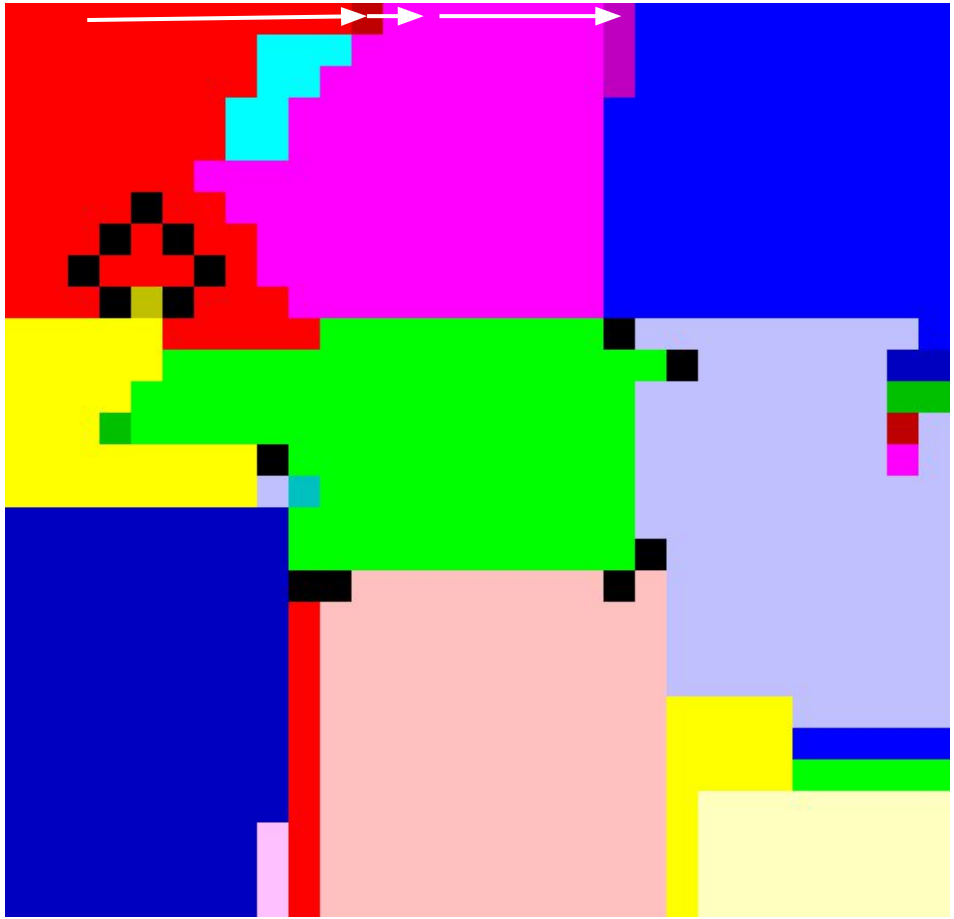
		Lightness change		
		None	**1 darker**	**2 darker**
Hue change	**None**		PUSH	POP
	1 step	ADD	SUB	MUL
	2 steps	DIV	MOD	NOT
	3 steps	>	POINTER	SWITCH
	4 steps	DUP	ROLL	IN(num)
	5 steps	IN(char)	OUT(num)	OUT(char)

```
PUSH 72
OUT(char) → H
```



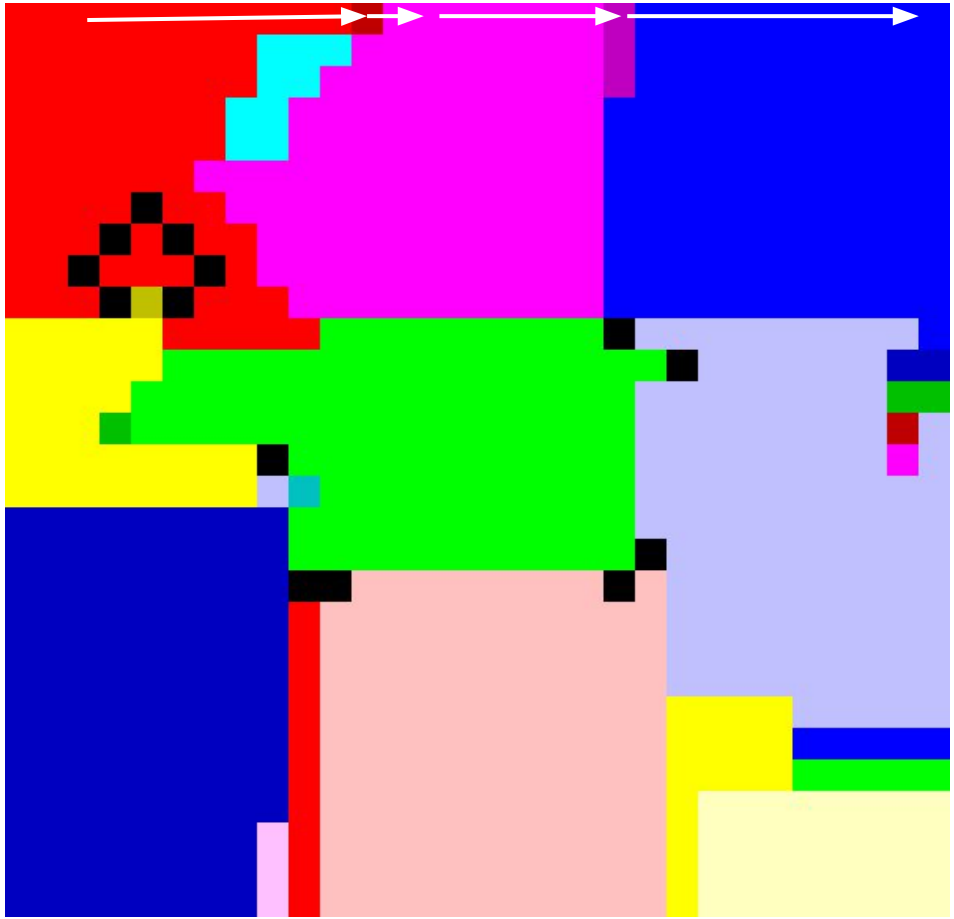
		Lightness change		
		None	1 darker	2 darker
Hue change	None		PUSH	POP
	1 step	ADD	SUB	MUL
	2 steps	DIV	MOD	NOT
	3 steps	>	POINTER	SWITCH
	4 steps	DUP	ROLL	IN(num)
	5 steps	IN(char)	OUT(num)	OUT(char)

```
PUSH 72
OUT(char) → H
PUSH 101
```



		Lightness change		
		None	**1 darker**	**2 darker**
Hue change	**None**		PUSH	POP
	1 step	ADD	SUB	MUL
	2 steps	DIV	MOD	NOT
	3 steps	>	POINTER	SWITCH
	4 steps	DUP	ROLL	IN(num)
	5 steps	IN(char)	OUT(num)	OUT(char)

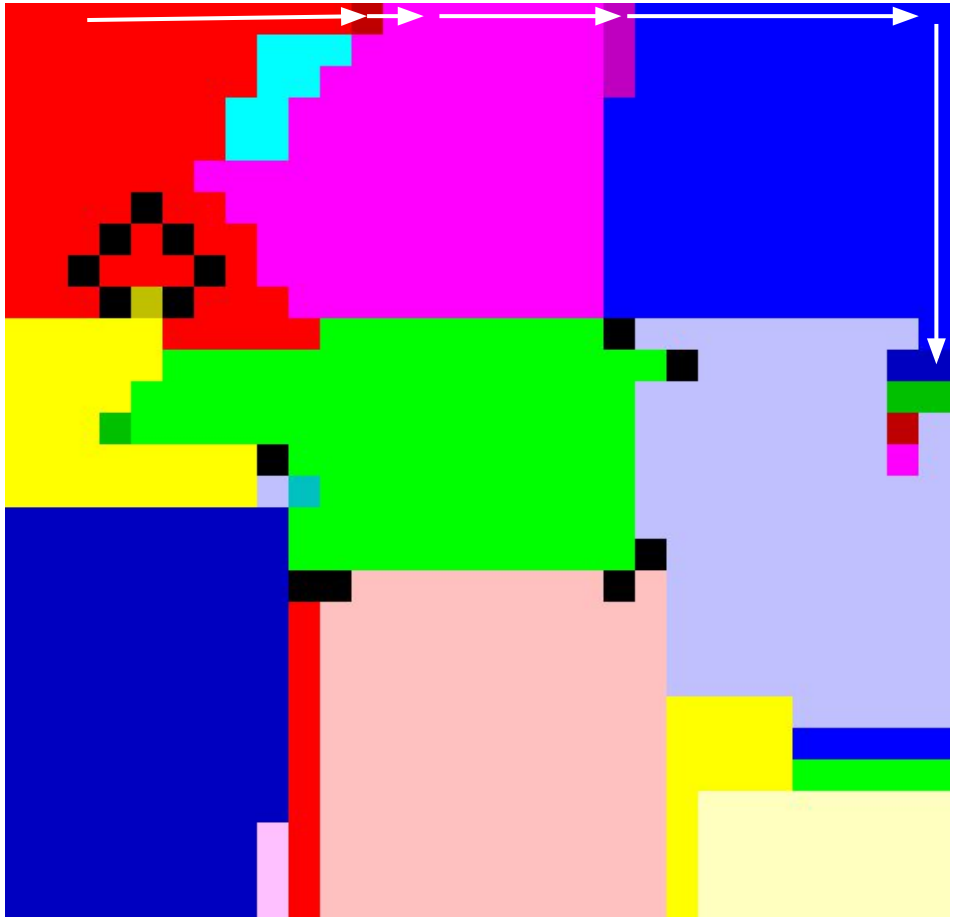
```
PUSH 72
OUT(char) → H
PUSH 101
OUT(char) → e
```



Rotates direction after hitting an edge

		Lightness change		
		None	**1 darker**	**2 darker**
Hue change	**None**		PUSH	POP
	1 step	ADD	SUB	MUL
	2 steps	DIV	MOD	NOT
	3 steps	>	POINTER	SWITCH
	4 steps	DUP	ROLL	IN(num)
	5 steps	IN(char)	OUT(num)	OUT(char)

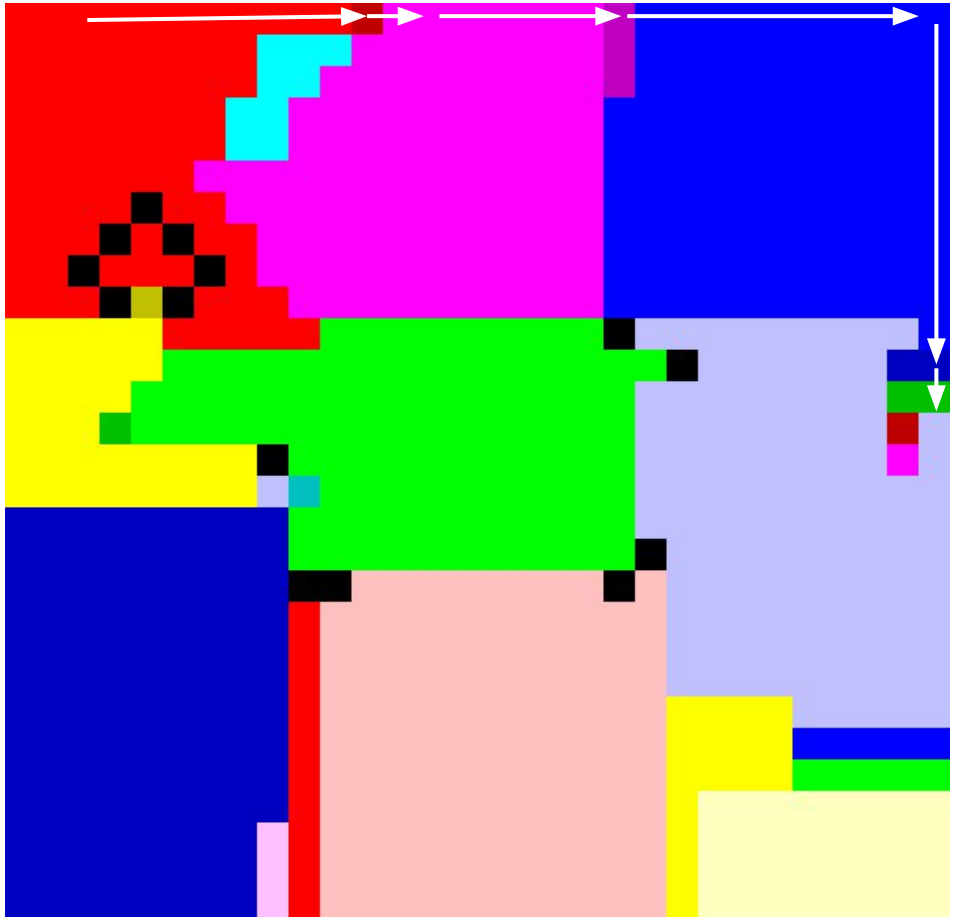
```
PUSH 72
OUT(char) → H
PUSH 101
OUT(char) → e
PUSH 108
```



Continues moving down...

		Lightness change		
		None	**1 darker**	**2 darker**
Hue change	**None**		PUSH	POP
	1 step	ADD	SUB	MUL
	2 steps	DIV	MOD	NOT
	3 steps	>	POINTER	SWITCH
	4 steps	DUP	ROLL	IN(num)
	5 steps	IN(char)	OUT(num)	OUT(char)

```
PUSH 72
OUT(char) → H
PUSH 101
OUT(char) → e
PUSH 108
DUP
```



The interpreter may not travel through every block in the image, so we can put whatever we want in those areas...

The interpreter passes around the border of the image and stops...

So we can put whatever we want in the centre

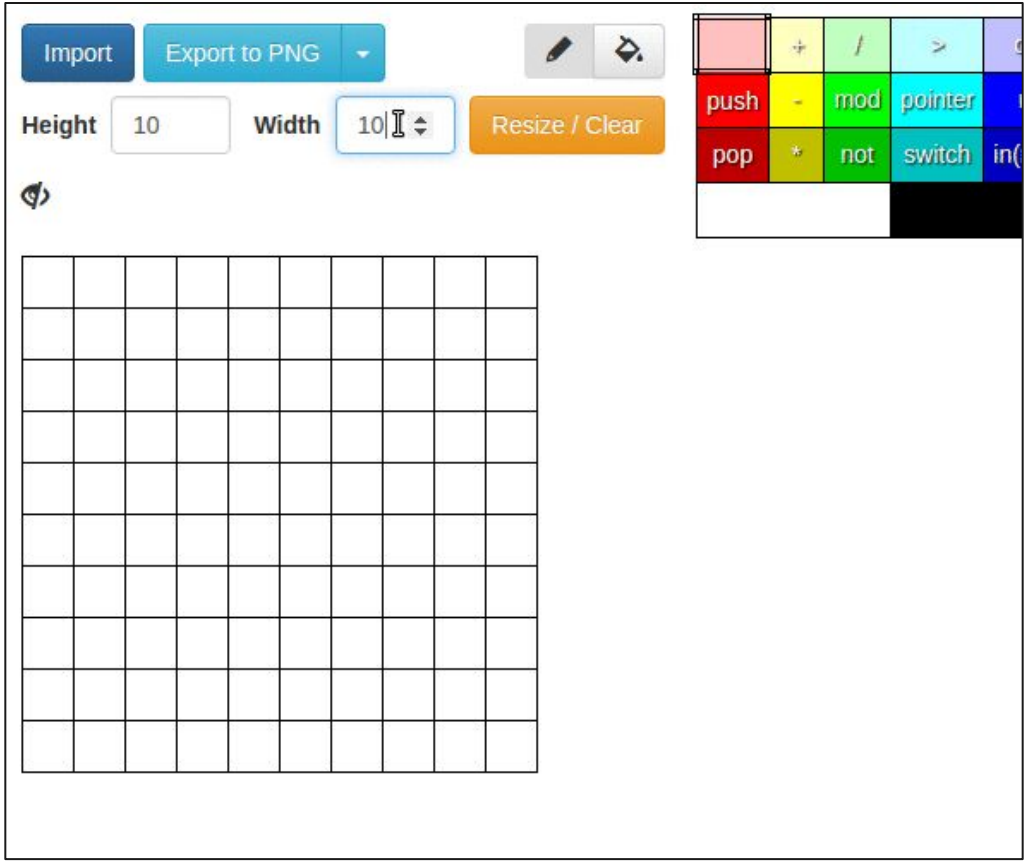
Hello World 6

Developing with MasterPiets

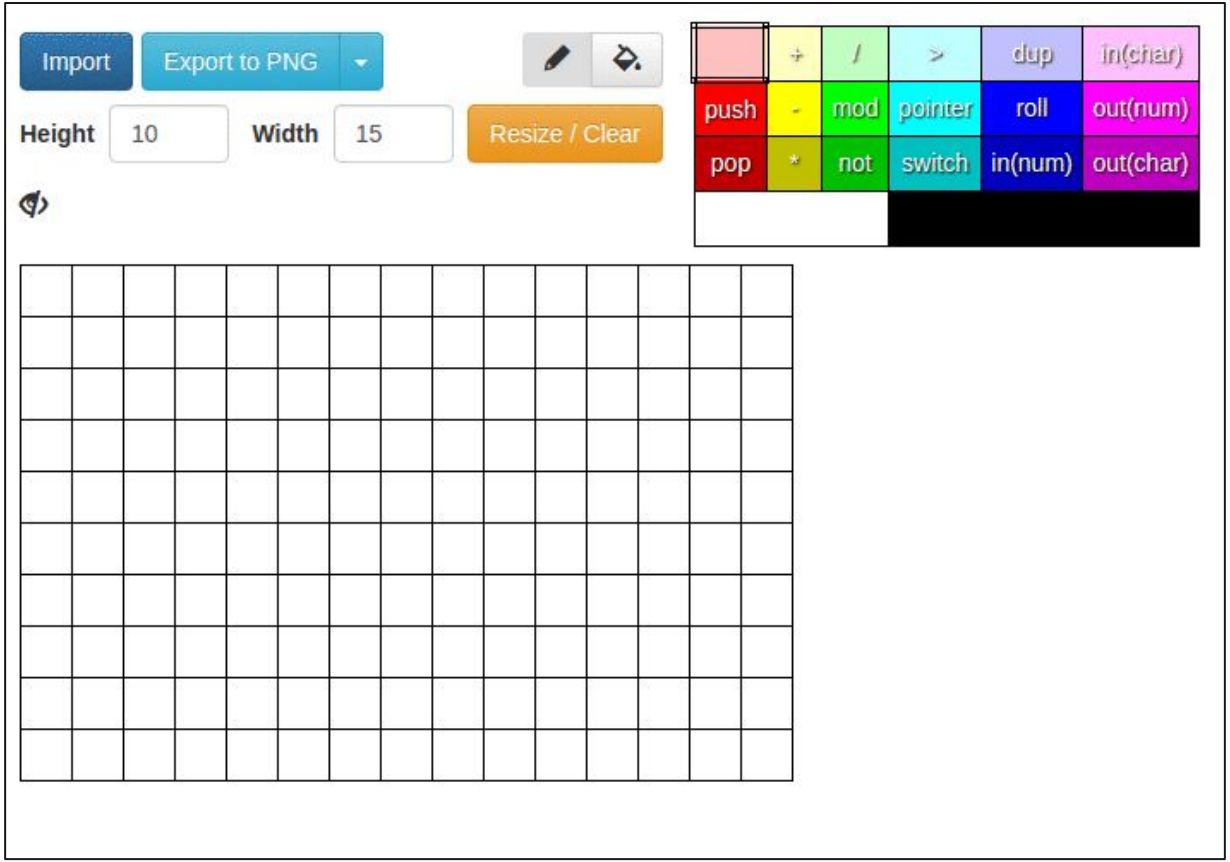
Many Piet-specific development tools exist (of course, you can create Piet programs with graphics software as well).

I created MasterPiets because I wanted a Piet IDE that:

- Was **easy/quick to start using** (incl. not needing to be downloaded)
- Had a simple and elegant **interface**
- Was **powerful** enough to manipulate large images
- Included a **debugger** with standard capabilities

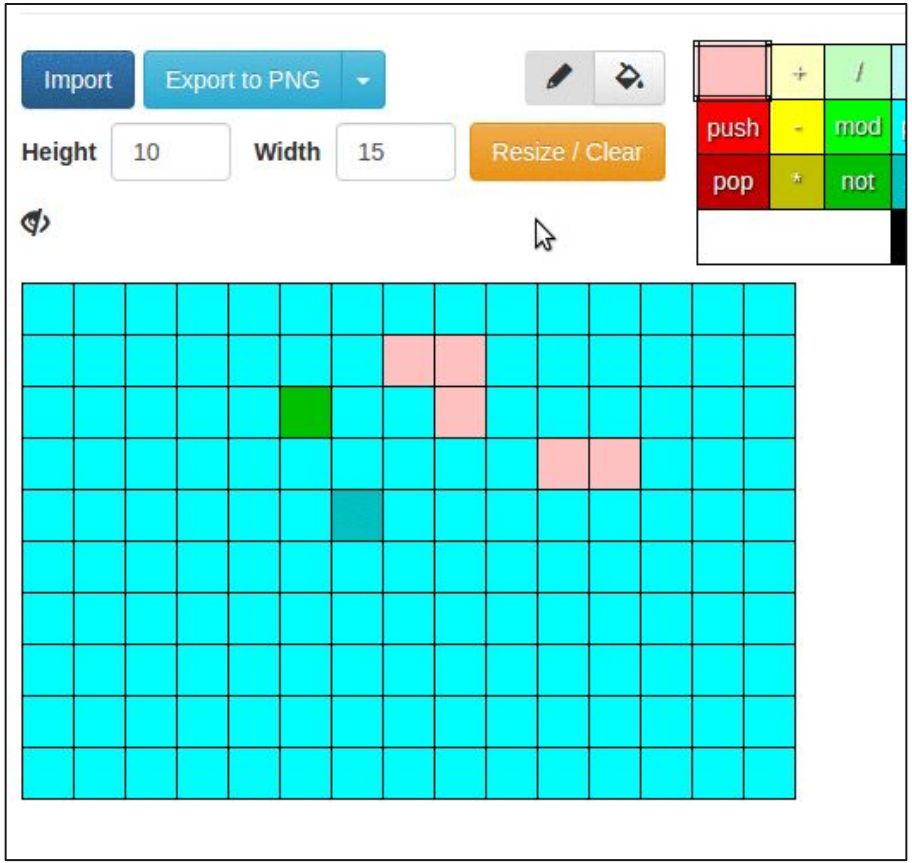


Choose the image size

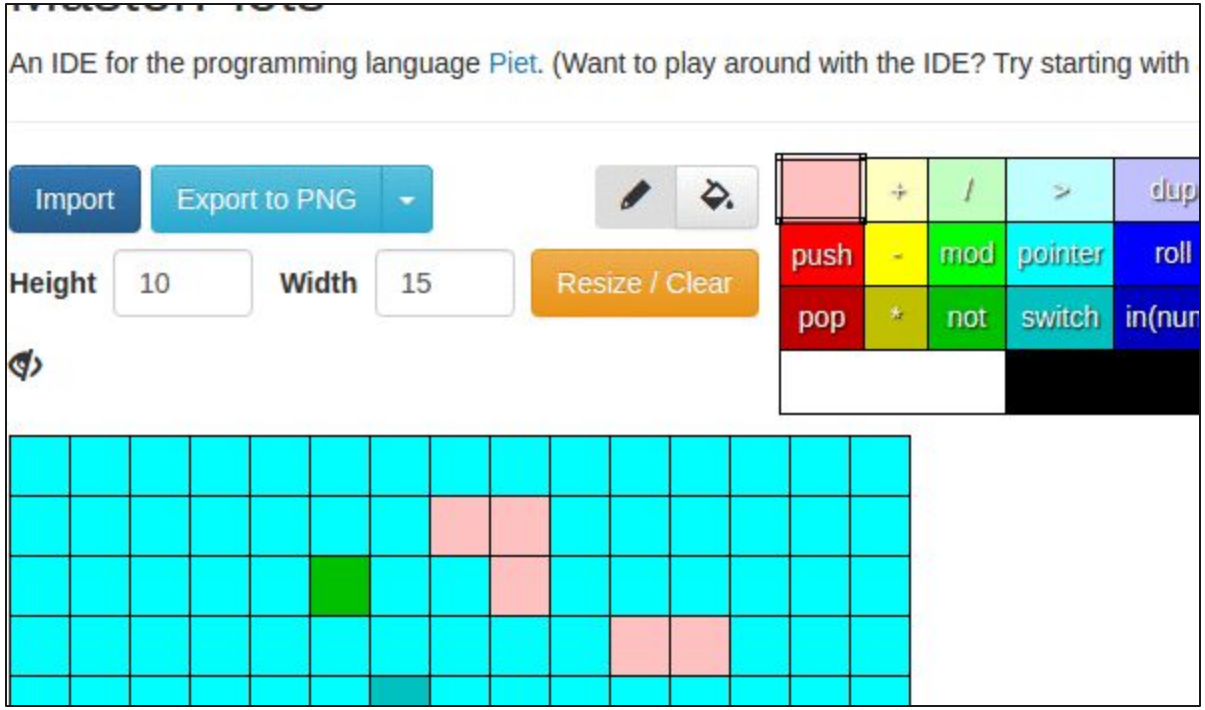


For each colour, the command is displayed that corresponds to a transition from the selected colour

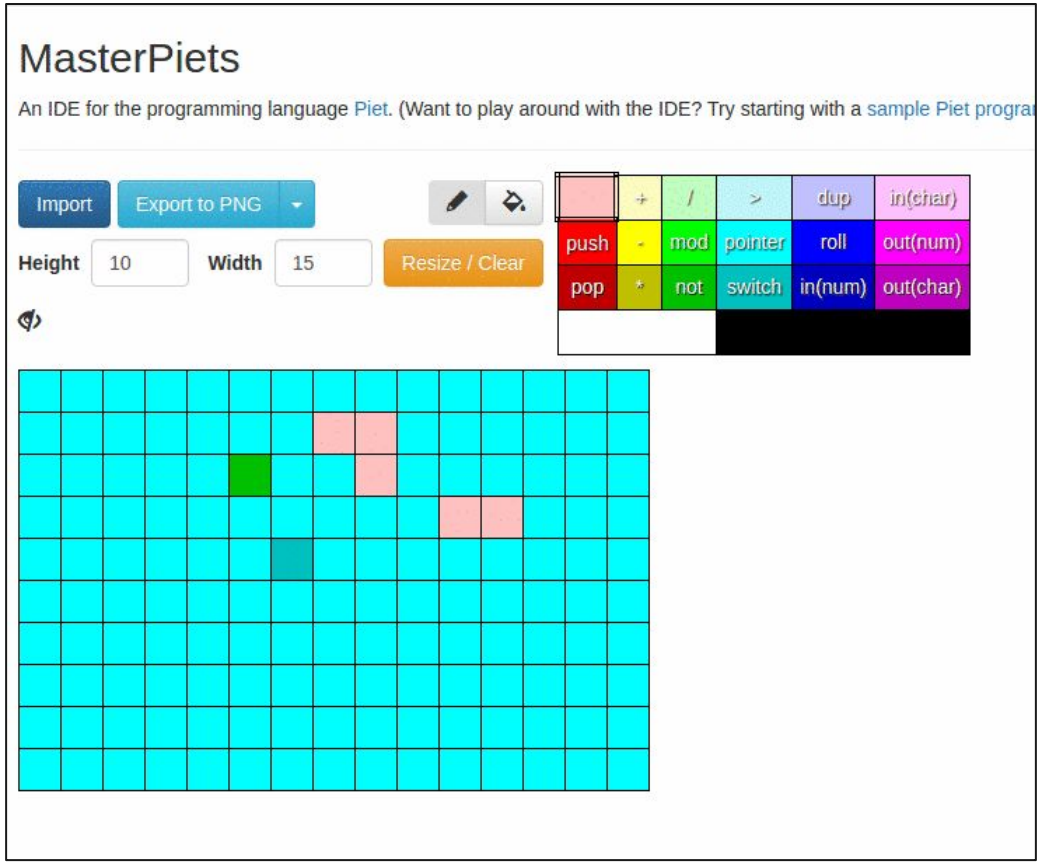
Paint individual pixels or an entire block



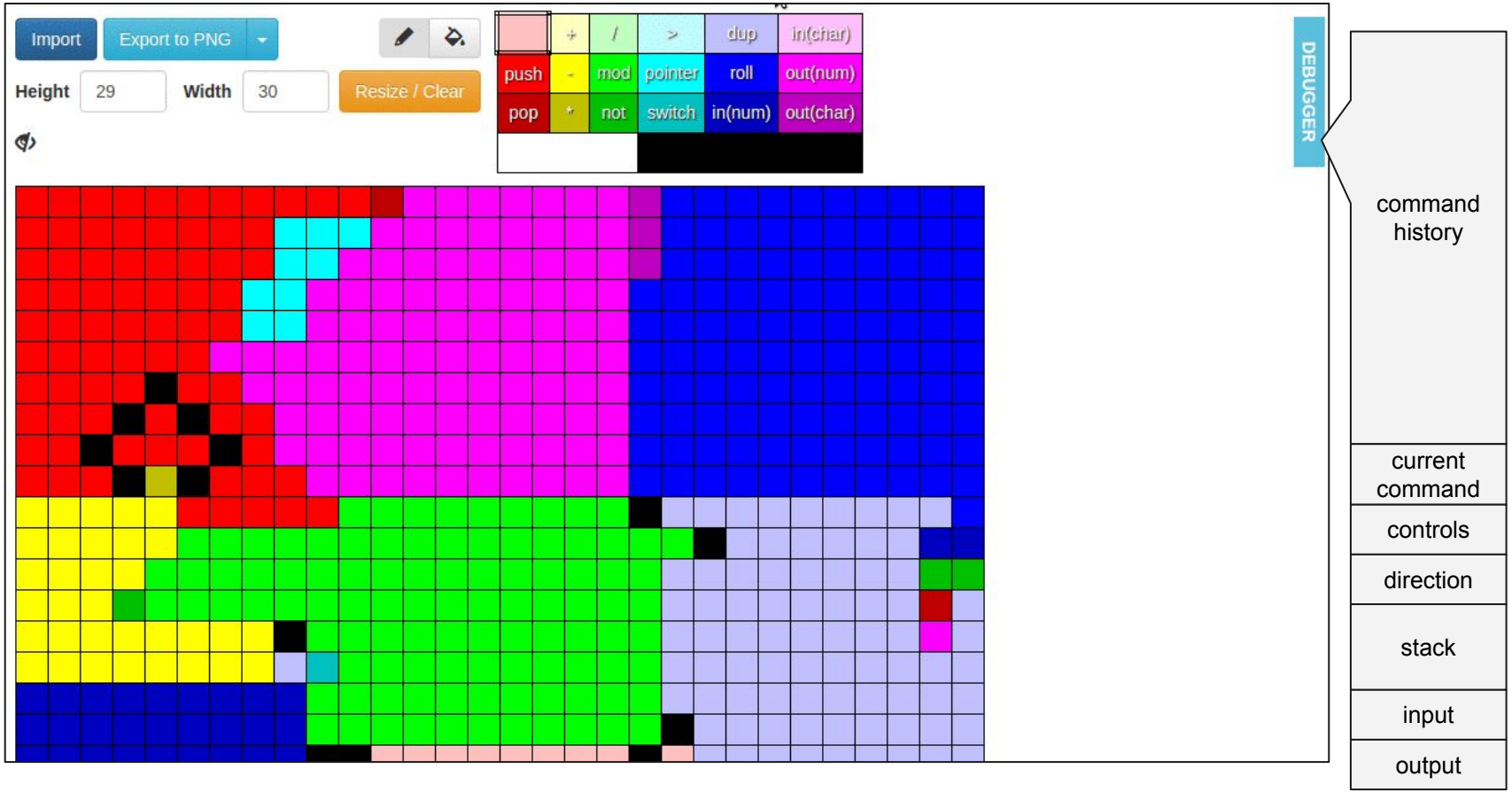
View the pixel count for each block, or make all pixel counts visible



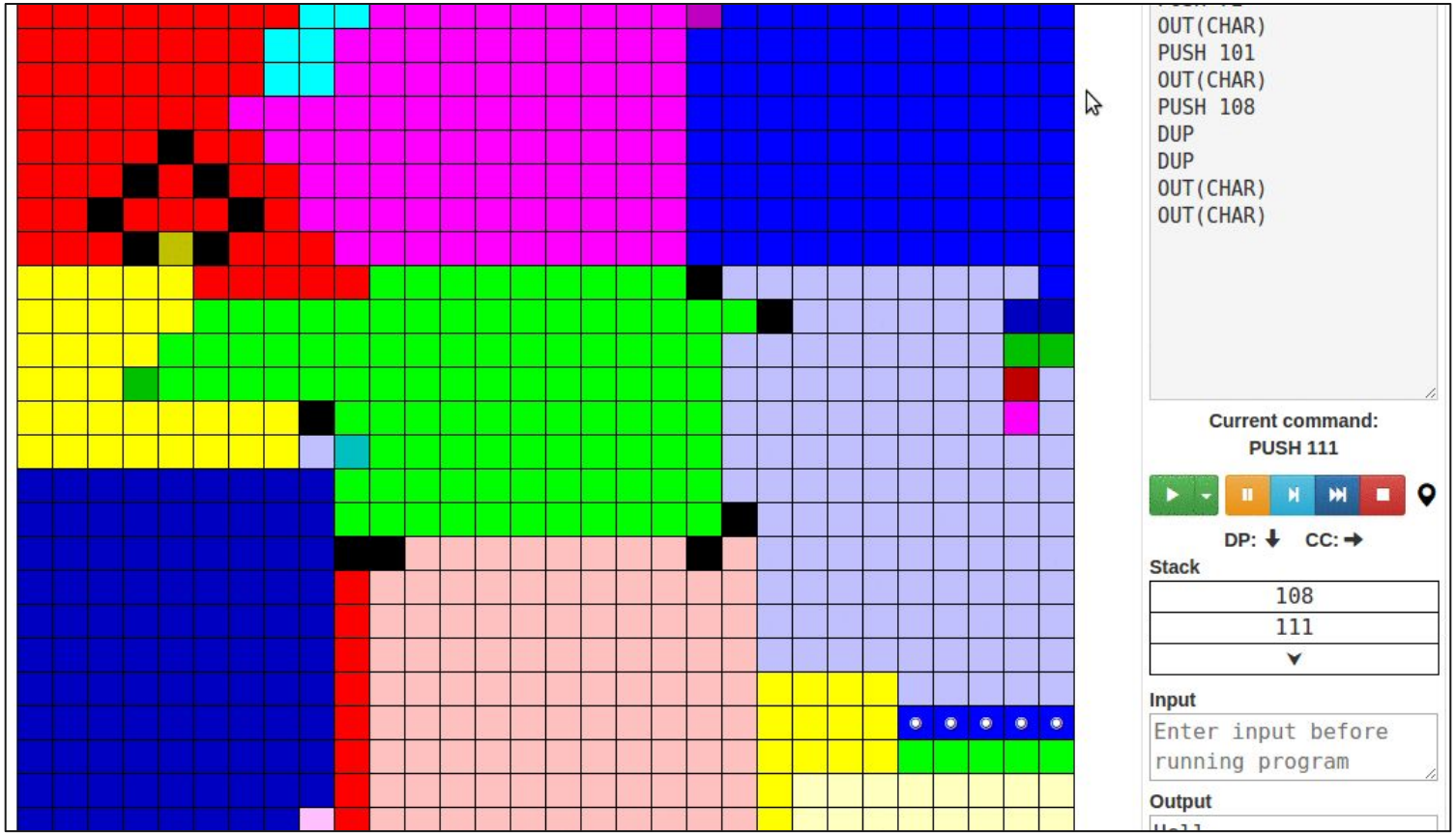
Export a program to a file (can be scaled up to show details)



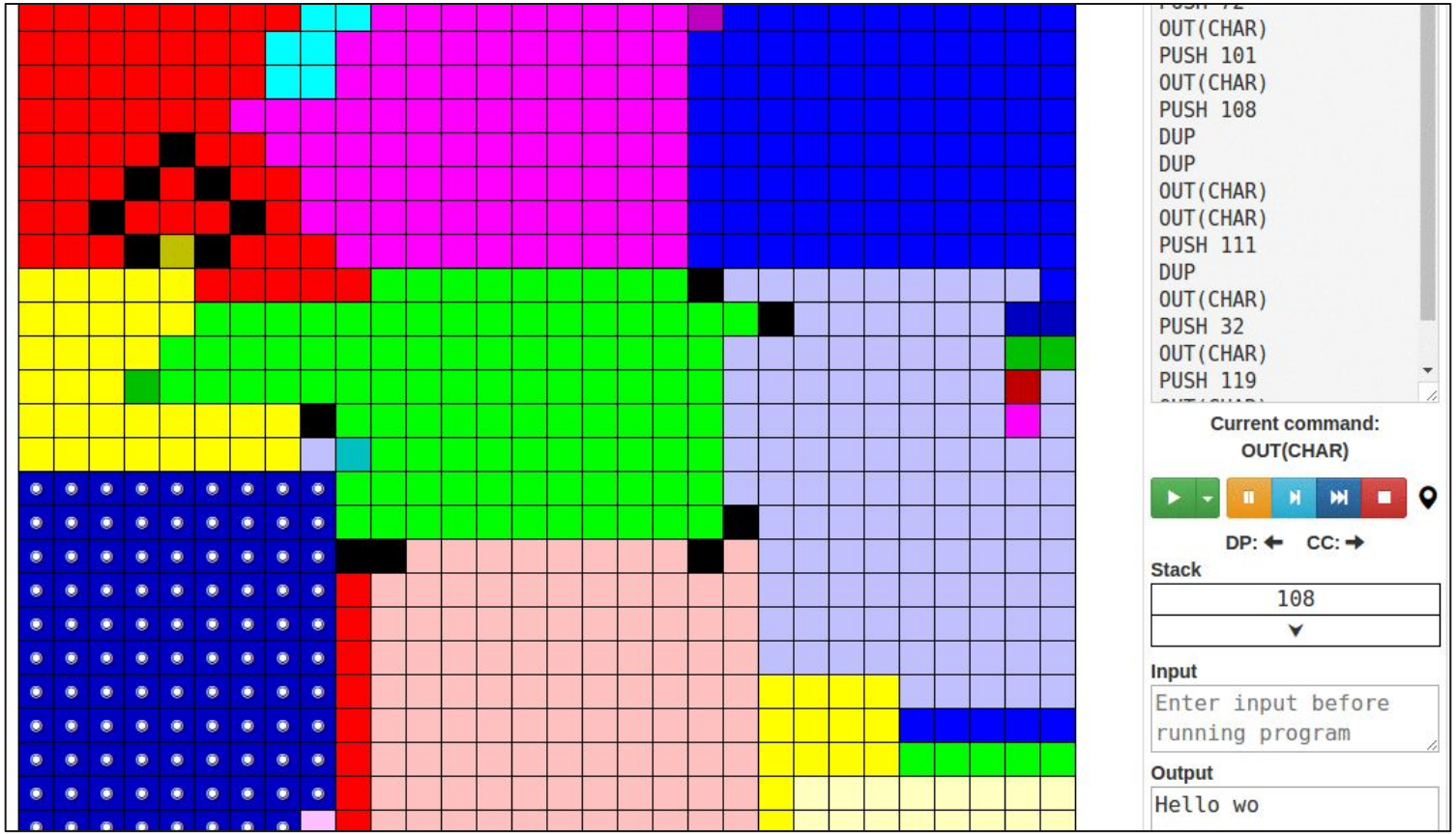
Import a program from a file



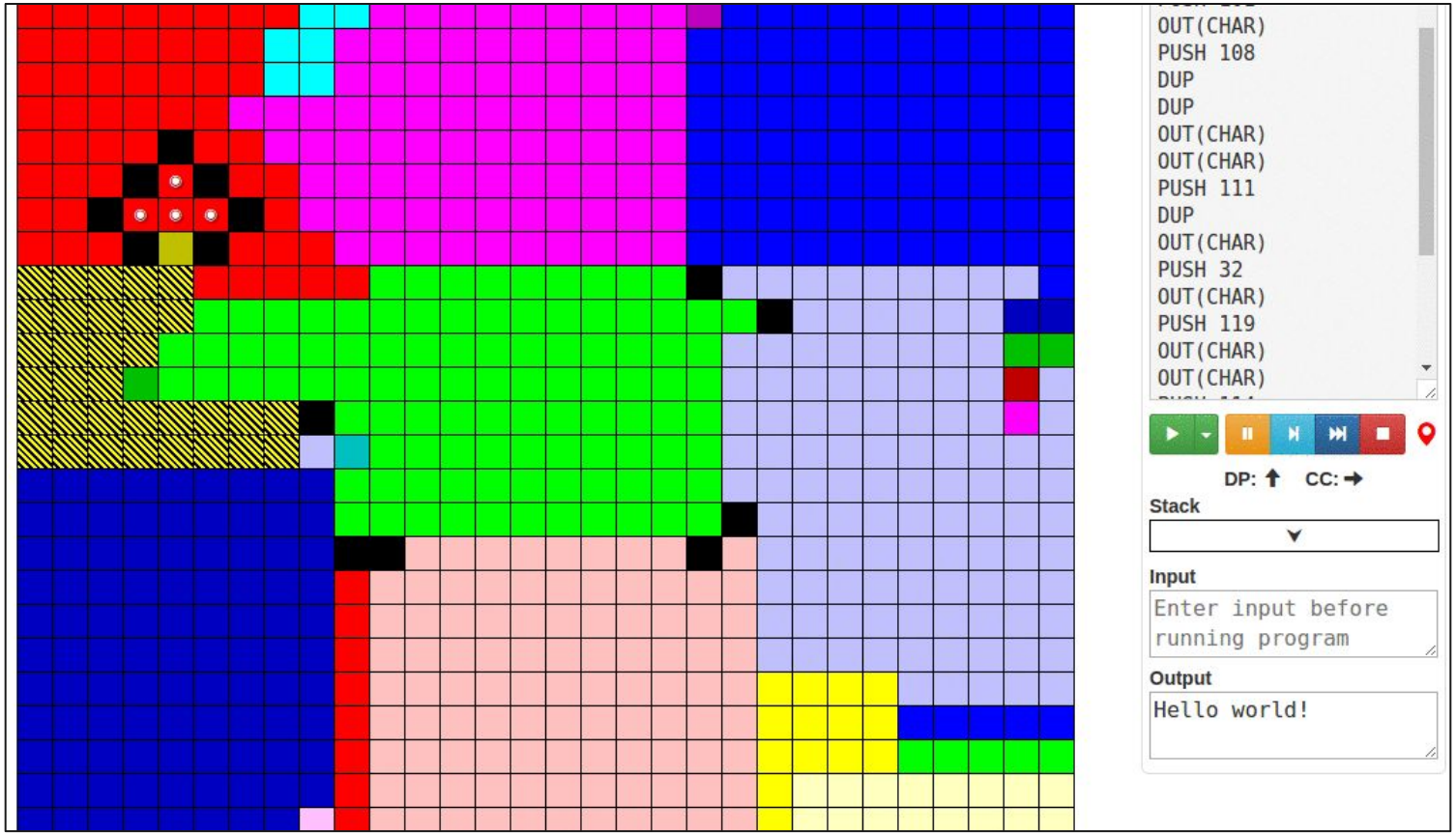
Run the program and watch the current state of the interpreter change



Pause the interpreter / Step through the program



Set a breakpoint on a block



Look back through the command history to trace the interpreter's path

Architecture

- Written in JavaScript, using React.js for the UI
- Runs entirely in the browser
- Uses Jimp for image importing/exporting

Next step

- Image recognition of a Piet program from a photograph (of a drawing, painting, etc.)