

The Bitmap Editor is used for creating graphical images using a visual “what you see is what you get” editor, and then automatically generating code that renders the images in the Hack or Jack programming languages. This can be useful when writing computer games and other programs that involve graphics and animation.

Images are 2D arrangements of pixels. In the low-level Hack assembly language, images can be created by writing 16-bit values to the screen memory. This process is tedious and error-prone.

In the high-level Jack language, images can be created using graphics OS functions like `drawPixel`, `drawLine`, `drawRectangel`, and `drawCircle`. For simple images, this solution may be acceptable. For complex images, and especially for images that must be rendered on the screen efficiently, a better solution is bypassing the OS by writing Jack code that writes the image data directly into the screen memory. The image data can be generated from the image by the bitmap editor.

Creating an image

Use the 2D grid to turn pixels on and off, until you’re happy with the image you’ve created. Then select one of the “generate Hack code” or “generate Jack code” options.

Using the generated code

Hack program: Copy-paste the generated code into your symbolic Hack program. To render the image on the screen, write Hack code that goes to the “draw” label, and then returns to the calling site in your code. If needed, change the automatically generated “draw” label to some other label name.

Jack code: Copy-paste the generated “draw” function into your Jack program. To render the image on the screen, write Jack code that calls the “draw” function. If needed, change the automatically generated “draw” name to some other name. Note that there is no need to use the draw function as is. Instead, you can copy-paste the generated code in the function’s body into one of the functions or methods in your Jack program, as needed.