More Loading an Image

The first part of the sample program (in `main.cpp`), loads a JPEG version of OCC's mascot, Pete the Pirate, into memory.

```
1  int width, height, bpp, channels = 4;
2  unsigned char * const pete =
3      stbi_load("pete.jpg",             // input file
4                &width, &height, &bpp,  // pointers (out)
5                channels);              // channels (in)
```

- The ***`stbi_load()`*** function returns a **pointer** to the first byte of the image data in memory. The type of the pointer is an **`unsigned char`**, which, in C++ speak means a "raw" byte. If loading fails, then the function returns the **`nullptr`**.

  Note that the pointer **`pete`** is a **`const`** pointer. This is necessary because you will later need to "free" the memory that the function has placed on the heap. If you move the pointer, then your program may crash.

- The first argument to the function is the **path to the file.** This can be absolute or relative (as used here), but it must be a **C-style string**. We'll look more at C-style strings in a future lesson. For right now, **use a literal** or use the **`c_str()`** member function on a regular C++ string.
- The next three arguments are the **address of the** **width**, **height**, and **bytes-per-pixel** used in the original image. These are **output parameters**; that means that you first create the variables (on line 1), and then pass **their addresses** as arguments. The function will **fill them in**. The information flows **out of the function**, not into it.
- The last argument is an **input** parameter telling the ***`stbi_load()`*** function how to store the image. Here we're telling it to store **4** bytes for each pixel (**RGBA**), even though the original image only has **3** (**RBG**).