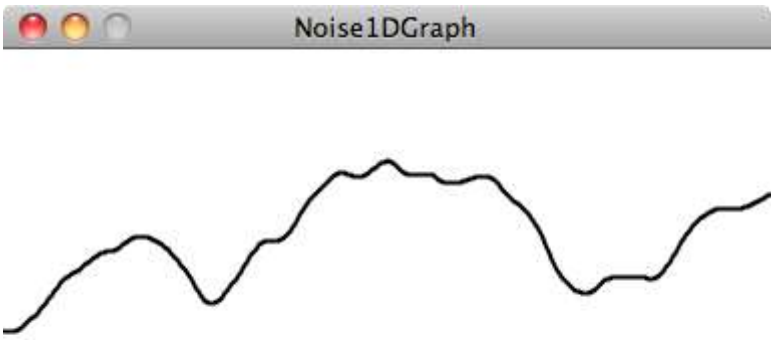
Perlin noise

A good random number generator produces numbers that have no relationship and show no discernible pattern. As we are beginning to see, a little bit of randomness can be a good thing when programming organic, lifelike behaviours. However, randomness as the single guiding principle is not necessarily natural. An algorithm known as “Perlin noise,” named for its inventor Ken Perlin, takes this concept into account. Perlin developed the noise function while working on the original Tron movie in the early 1980s; it was designed to create procedural textures for computer-generated effects. In 1997 Perlin won an Academy Award in technical achievement for this work. Perlin noise can be used to generate various effects with natural qualities, such as clouds, landscapes, and patterned textures like marble.

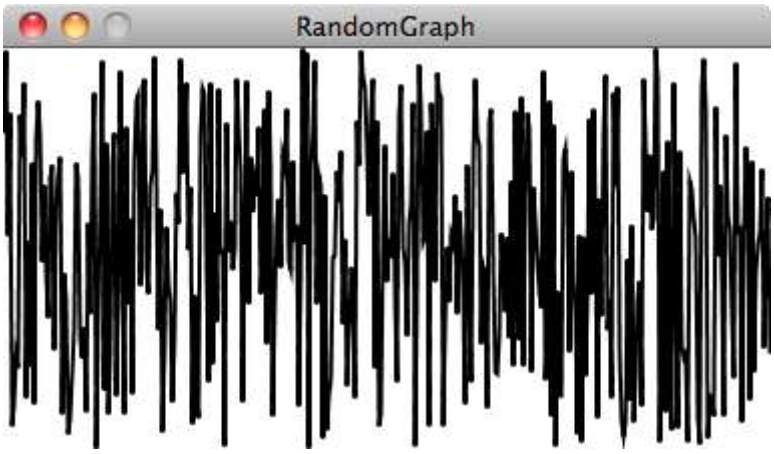
Perlin noise has a more organic appearance because it produces a naturally ordered (“smooth”) sequence of pseudo-random numbers. The graph below shows Perlin noise over time, with the x-axis representing time; note the smoothness of the curve.



Nature of Code Image

Figure I.5: Noise

Contrastingly, the next graph below shows pure random numbers over time.



Nature of Code Image

Figure I.6: Random

ProcessingJS has a built-in implementation of the Perlin noise algorithm: the function `noise()`. The `noise()` function takes one, two, or three arguments, as noise is computed in one, two, or three dimensions. Let’s start by looking at one-dimensional noise.