In Chapter 3, we churned out a full design for Mr. Nom, consisting of the game mechanics, a simple background story, handcrafted graphical assets, and definitions for all the screens based on some paper cutouts. In Chapter 5, we developed a full-fledged game-development framework that allows us to transfer our design screens easily to code. But enough talking; let’s start writing our first game!

Creating the Assets

We have two types of assets in Mr. Nom: audio assets and graphical assets. We recorded the audio assets via a nice open source application called Audacity and a bad netbook microphone. We created a sound effect to be played when a button is pressed or a menu item is chosen, one for when Mr. Nom eats a stain, and one for when he eats himself. We saved them as OGGs to the assets/ folder, under the names click.ogg, eat.ogg, and bitten.ogg, respectively. You can either be creative and create those files yourself using Audacity and a microphone, or you can fetch them from the SVN repository at http://code.google.com/p/beginningandroidgames2/.

See the front matter where we describe how to get the source code if you are unfamiliar with SVN.

Earlier, we mentioned that we’ll want to reuse those paper cutouts from the design phase as our real game graphics. For this, we first have to make them fit with our target resolution.

We chose a fixed target resolution of 320 × 480 (portrait mode) for which we’ll design all our graphic assets. This might seem small, but it made it very quick and easy for us to develop the game and graphics and, after all, the point here is that you get to see the entire Android game development process.

For your production game, consider all of the resolutions and use higher-resolution graphics so that your game looks good on tablet-sized screens, perhaps targeting 800 × 1280 as a baseline. We scanned in all the paper cutouts and resized them a bit. We saved most of the assets in separate files and merged some of them into a single file. All images are saved in a PNG format. The background is the only image that is RGB888; all others are ARGB8888. Figure 6-1 shows you what we ended up with.
Let's break down those images a little:

background.png: This is our background image, which will be the first thing we'll draw to the framebuffer. It has the same size as our target resolution for obvious reasons.

buttons.png: This contains all the buttons we'll need in our game. We put them into a single file, as we can easily draw them via the Graphics.drawPixmap() method, which allows drawing portions of an image. We'll use that technique more often when we start drawing with OpenGL ES, so we better get used to it now. Merging several images into a single image is often called **atlasing**, and

Figure 6-1. All the graphic assets of Mr. Nom with their respective filenames and sizes in pixels

Let's break down those images a little:

background.png: This is our background image, which will be the first thing we’ll draw to the framebuffer. It has the same size as our target resolution for obvious reasons.

buttons.png: This contains all the buttons we’ll need in our game. We put them into a single file, as we can easily draw them via the Graphics.drawPixmap() method, which allows drawing portions of an image. We’ll use that technique more often when we start drawing with OpenGL ES, so we better get used to it now. Merging several images into a single image is often called **atlasing**, and