This function accepts the `music_path` variable and makes a list of all the MP3 files that are present. It then shuffles the music list using the `random.shuffle()` function. Doing so ensures that the order in which the songs are played is different each time. Now you iterate through the shuffled list and play the music files one by one. If there is an `IndexError` exception, it passes on the message "No music files found."

Make the following changes in `brain.py` and add this code snippet:

```
elif check_message(['party', 'time']) or check_message(['party', 'mix']):
    play_music.play_shuffle(music_path)
```

This ensures that the party mix is called when the user says something like, "It's party time!" or "Party mix!" Note that you must add this code snippet above the time module's code snippet. If you don't, then if you say "It's party time!," the software will detect the *time* keyword, and the logical engine will transfer control of the program to the time module.

Summary

In this chapter, you learned how to build music-player functionality for Melissa, and you learned to find all the MP3 files in a folder and its subfolders, make a list of them, play them randomly, search for specific music, and create a party mix. Did I forget to mention that you made all of this voice controlled?

In the next chapter, you build a voice-controlled note-taking application that lets you save notes using your voice and retrieve them later.