Introduction

Pro Django represents seven years of accumulated knowledge in Python and Django, designed to educate readers who are already familiar with both topics and would like to take them further than they had previously done. You will learn a wide range of advanced techniques available in both Python and Django, along with tips on how to use them to achieve advanced functionality.

This book is designed to be both a narrative to be read from start to finish and a general reference to be searched for specific information. Since you may not know what to look for or where to find it yet, feel free to read through the book first, then keep it handy for refreshing your memory as necessary.

What This Book Is Not

There are plenty of resources available for learning Python and Django, so this book does not strive to teach the basics. For readers new to Python, I highly recommend *Dive Into Python 3* by Mark Pilgrim (Apress, 2009). For learning Django, I'd recommend *The Definitive Guide to Django: Web Development Done Right* by Adrian Holovaty and Jacob Kaplan-Moss (Second Edition, Apress, 2009). Additionally, *Practical Django Projects* by James Bennett (Second Edition, Apress, 2009) is an excellent resource for general application development.

Who This Book Is For

Because *Pro Django* doesn't dwell on introductory details, readers will be expected to have experience with both Python and Django. If you're new to either subject, please consider one of the books mentioned in the previous section before trying to tackle this book.

Even if you've only experimented on your own without launching a full site yet, a basic familiarity should be sufficient. You don't need to be an expert to start reading *Pro Django*, but you might be by the time you finish.

Interpreting Code Samples

Pro Django uses a simple format, interleaving explanations of Python's and Django's available features with code that demonstrates their use in the real world. There are two types of code samples used, which differ in how they should be executed.

Python's interactive interpreter is a great way to test out small pieces of code and see how it works in a variety of situations. Lines of code intended for use in that environment will always be prefixed with three characters: three greater-than signs (`>>>`) or three periods (`...`). Lines with greater-than signs are the outermost block of code, while the period-prefixed lines are indented at least one level. The three initial characters are also followed by a space. These first four characters are not typed into the interactive interpreter directly; they simply mimic what the interpreter itself looks like by reproducing its output.