Test-Driven Development with Python

Harry Percival
## Table of Contents

Preface .......... xlv
Prerequisites and Assumptions .......... xxi
Acknowledgments .......... xxvii

### Part I. The Basics of TDD and Django

1. Getting Django Set Up Using a Functional Test. 3
   - Obey the Testing Goat! Do Nothing Until You Have a Test 3
   - Getting Django Up and Running 6
   - Starting a Git Repository 8

2. Extending Our Functional Test Using the unittest Module. 13
   - Using a Functional Test to Scope Out a Minimum Viable App 13
   - The Python Standard Library’s unittest Module 16
   - Implicit waits 18
   - Commit 18

3. Testing a Simple Home Page with Unit Tests. 21
   - Our First Django App, and Our First Unit Test 22
   - Unit Tests, and How They Differ from Functional Tests 22
   - Unit Testing in Django 23
   - Django’s MVC, URLs, and View Functions 24
   - At Last! We Actually Write Some Application Code! 26
   - urls.py 27
   - Unit Testing a View 30
     - The Unit-Test/Code Cycle 31

4. What Are We Doing with All These Tests? 35
   - Programming Is like Pulling a Bucket of Water up from a Well 36
Using Selenium to Test User Interactions .......................................................... 37
The "Don't Test Constants" Rule, and Templates to the Rescue .......................... 40
Refactoring to Use a Template ............................................................................ 40
On Refactoring ...................................................................................................... 44
A Little More of Our Front Page ........................................................................... 45
Recap: The TDD Process .................................................................................... 47

5. Saving User Input ............................................................................................. 51
Wiring Up Our Form to Send a POST Request .................................................. 51
Processing a POST Request on the Server ....................................................... 54
Passing Python Variables to Be Rendered in the Template .............................. 55
Three Strikes and Refactor .................................................................................. 59
The Django ORM and Our First Model ............................................................... 60
Our First Database Migration .............................................................................. 62
The Test Gets Surprisingly Far .......................................................................... 63
A New Field Means a New Migration ................................................................. 64
Saving the POST to the Database ...................................................................... 65
Redirect After a POST .......................................................................................... 68
Better Unit Testing Practice: Each Test Should Test One Thing ...................... 68
Rendering Items in the Template ....................................................................... 69
Creating Our Production Database with migrate .......................................... 71

6. Getting to the Minimum Viable Site ............................................................... 77
Ensuring Test Isolation in Functional Tests ....................................................... 77
Running Just the Unit Tests ................................................................................ 80
Small Design When Necessary .......................................................................... 81
YAGNI! .................................................................................................................. 82
REST ..................................................................................................................... 82
Implementing the New Design Using TDD .......................................................... 83
Iterating Towards the New Design ...................................................................... 86
Testing Views, Templates, and URLs Together with the Django Test Client ........ 87
A New Test Class ................................................................................................. 88
A New URL .......................................................................................................... 88
A New View Function ........................................................................................ 89
A Separate Template for Viewing Lists ............................................................. 90
Another URL and View for Adding List Items ................................................... 92
A Test Class for New List Creation .................................................................... 93
A URL and View for New List Creation ............................................................. 94
Removing Now-Redundant Code and Tests ....................................................... 95
Pointing Our Forms at the New URL ................................................................. 96
Adjusting Our Models ........................................................................................ 97
A Foreign Key Relationship ............................................................................... 99
### Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adjusting the Database Location</td>
<td>141</td>
</tr>
<tr>
<td>Creating a Virtualenv</td>
<td>142</td>
</tr>
<tr>
<td>Simple Nginx Configuration</td>
<td>144</td>
</tr>
<tr>
<td>Creating the Database with migrate</td>
<td>147</td>
</tr>
<tr>
<td>Getting to a Production-Ready Deployment</td>
<td>148</td>
</tr>
<tr>
<td>Switching to Gunicorn</td>
<td>148</td>
</tr>
<tr>
<td>Getting Nginx to Serve Static Files</td>
<td>149</td>
</tr>
<tr>
<td>Switching to Using Unix Sockets</td>
<td>150</td>
</tr>
<tr>
<td>Switching <code>DEBUG</code> to False and Setting <code>ALLOWED_HOSTS</code></td>
<td>151</td>
</tr>
<tr>
<td>Using Upstart to Make Sure Gunicorn Starts on Boot</td>
<td>151</td>
</tr>
<tr>
<td>Saving Our Changes: Adding Gunicorn to Our <code>requirements.txt</code></td>
<td>152</td>
</tr>
<tr>
<td>Automating</td>
<td>152</td>
</tr>
<tr>
<td>“Saving Your Progress”</td>
<td>156</td>
</tr>
<tr>
<td>9. Automating Deployment with Fabric</td>
<td>157</td>
</tr>
<tr>
<td>Breakdown of a Fabric Script for Our Deployment</td>
<td>158</td>
</tr>
<tr>
<td>Trying It Out</td>
<td>162</td>
</tr>
<tr>
<td>Deploying to Live</td>
<td>163</td>
</tr>
<tr>
<td>Nginx and Gunicorn Config Using <code>sed</code></td>
<td>165</td>
</tr>
<tr>
<td>Git Tag the Release</td>
<td>166</td>
</tr>
<tr>
<td>Further Reading</td>
<td>166</td>
</tr>
<tr>
<td>10. Input Validation and Test Organisation</td>
<td>169</td>
</tr>
<tr>
<td>Validation FT: Preventing Blank Items</td>
<td>169</td>
</tr>
<tr>
<td>Skipping a Test</td>
<td>170</td>
</tr>
<tr>
<td>Splitting Functional Tests out into Many Files</td>
<td>171</td>
</tr>
<tr>
<td>Running a Single Test File</td>
<td>174</td>
</tr>
<tr>
<td>Flesching Out the FT</td>
<td>174</td>
</tr>
<tr>
<td>Using Model-Layer Validation</td>
<td>175</td>
</tr>
<tr>
<td>Refactoring Unit Tests into Several Files</td>
<td>175</td>
</tr>
<tr>
<td>Unit Testing Model Validation and the <code>self.assertRaises</code> Context Manager</td>
<td>177</td>
</tr>
<tr>
<td>A Django Quirk: Model Save Doesn't Run Validation</td>
<td>178</td>
</tr>
<tr>
<td>Surfacing Model Validation Errors in the View</td>
<td>178</td>
</tr>
<tr>
<td>Checking Invalid Input Isn't Saved to the Database</td>
<td>181</td>
</tr>
<tr>
<td>Django Pattern: Processing POST Requests in the Same View as Renders the Form</td>
<td>183</td>
</tr>
<tr>
<td>Refactor: Transferring the <code>new_item</code> Functionality into <code>view_list</code></td>
<td>184</td>
</tr>
<tr>
<td>Enforcing Model Validation in <code>view_list</code></td>
<td>186</td>
</tr>
<tr>
<td>Refactor: Removing Hardcoded URLs</td>
<td>187</td>
</tr>
<tr>
<td>The <code>{% url %}</code> Template Tag</td>
<td>188</td>
</tr>
</tbody>
</table>
11. A Simple Form

- Moving Validation Logic into a Form
- Exploring the Forms API with a Unit Test
- Switching to a Django ModelForm
- Testing and Customising Form Validation
- Using the Form in Our Views
  - Using the Form in a View with a GET Request
  - A Big Find and Replace
- Using the Form in a View That Takes POST Requests
  - Adapting the Unit Tests for the new_list View
  - Using the Form in the View
  - Using the Form to Display Errors in the Template
- Using the Form in the Other View
  - A Helper Method for Several Short Tests
- Using the Form's Own Save Method

12. More Advanced Forms

- Another FT for Duplicate Items
  - Preventing Duplicates at the Model Layer
  - A Little Digression on Queryset Ordering and String Representations
  - Rewriting the Old Model Test
  - Some Integrity Errors Do Show Up on Save
- Experimenting with Duplicate Item Validation at the Views Layer
- A More Complex Form to Handle Uniqueness Validation
- Using the Existing List Item Form in the ListView

13. Dipping Our Toes, Very Tentatively, into JavaScript

- Starting with an FT
- Setting Up a Basic JavaScript Test Runner
- Using jQuery and the Fixtures Div
- Building a JavaScript Unit Test for Our Desired Functionality
- Javascript Testing in the TDD Cycle
- Columbo Says: Onload Boilerplate and Namespacing
- A Few Things That Didn't Make It


- Staging Deploy
- Live Deploy
- What to Do If You See a Database Error
Part III. More Advanced Topics

15. User Authentication, Integrating Third-Party Plugins, and Mocking with JavaScript. 241
   Mozilla Persona (BrowserID) 242
   Exploratory Coding, aka “Spiking” 242
      Starting a Branch for the Spike 243
      Frontend and JavaScript Code 243
      The Browser-ID Protocol 244
      The Server Side: Custom Authentication 245
   De-spiking 251
      A Common Selenium Technique: Explicit Waits 253
      Reverting Our Spiked Code 255
   JavaScript Unit Tests Involving External Components: Our First Mocks! 256
      Housekeeping: A Site-Wide Static Files Folder 256
      Namespacing 258
      A Simple Mock to Unit Tests Our initialize Function 258
      More Advanced Mocking 264
      Checking Call Arguments 267
      QUnit setup and teardown, Testing Ajax 268
      More Nested Callbacks! Testing Asynchronous Code 272

16. Server-Side Authentication and Mocking in Python. 277
   A Look at Our Spiked Login View 277
   Mocking in Python 278
      Testing Our View by Mocking Out authenticate 278
      Checking the View Actually Logs the User In 281
   De-spiking Our Custom Authentication Backend: Mocking Out an Internet Request 285
      1 if = 1 More Test 286
      Patching at the Class Level 287
      Beware of Mocks in Boolean Comparisons 290
      Creating a User if Necessary 291
      The get_user Method 291
   A Minimal Custom User Model 293
      A Slight Disappointment 295
      Tests as Documentation 296
      Users Are Authenticated 297
      The Moment of Truth: Will the FT Pass? 298
Moving Down to the Forms Layer
Keep Listening to Your Tests: Removing ORM Code from Our Application
Finally, Moving Down to the Models Layer
Back to Views
The Moment of Truth (and the Risks of Mocking)
Thinking of Interactions Between Layers as “Contracts”
Identifying Implicit Contracts
Fixing the Oversight
One More Test
Tidy Up: What to Keep from Our Integrated Test Suite
Removing Redundant Code at the Forms Layer
Removing the Old Implementation of the View
Removing Redundant Code at the Forms Layer
Conclusions: When to Write Isolated Versus Integrated Tests
Let Complexity Be Your Guide
Should You Do Both?
Onwards!

20. Continuous Integration (CI) ......................................................... 365
   Installing Jenkins
   Configuring Jenkins Security
   Adding Required Plugins
Setting Up Our Project
First Build!
Setting Up a Virtual Display so the FTs Can Run Headless
Taking Screenshots
A Common Selenium Problem: Race Conditions
Running Our QUnit JavaScript Tests in Jenkins with PhantomJS
   Installing node
   Adding the Build Steps to Jenkins
More Things to Do with a CI Server

21. The Token Social Bit, the Page Pattern, and an Exercise for the Reader ............ 387
   An FT with Multiple Users, and addCleanup
   Implementing the Selenium Interact/Wait Pattern
   The Page Pattern
   Extend the FT to a Second User, and the “My Lists” Page
   An Exercise for the Reader

22. Fast Tests, Slow Tests, and Hot Lava .............................................. 397
   Thesis: Unit Tests Are Superfast and Good Besides That
   Faster Tests Mean Faster Development