

## Chapter 1

### Page 26

1. “Grails automatically created a shell unit-test case in `/grails-app/test/unit/QuoteServiceTests.groovy`” is wrong. Remove `“/grails-app”`
2. “This will tell Grails to create a shell `/grails-app/test/integration/QuoteServiceIntegrationTests.groovy` file.” Again remove `“/grails-app”`
3. In Listing 1.17 `class QuoteServiceTests extends GrailsUnitTestCase` should be `class QuoteServiceIntegrationTests extends GrailsUnitTestCase`
4. Listing 1.18 has same issue as Listing 1.17
5. The test fails because of difference in case in `assertEquals("Real Programmers Don't eat Quiche", staticQuote.content)` should be `assertEquals("Real Programmers Don't eat quiche", staticQuote.content)`

### Page 27

1. “Grails also generates an HTML version of our test results, which you can view by opening `/grails-app/test/reports/html/index.html` in a web browser.” Remove the `“/grails-app”`

### Page 28

1. In listing 1.22, `render "...` should be replaced by `response.outputStream << "...`. This is because the `render()` method applies the `quote.gsp` layout to the generated output, which isn't what we want.

## Chapter 3

### Page 77

1. In the table, the example for `inList` is missing a closing `']`. It should be `country(inList: ['Australia', 'England'])`

Chapter author name:

©Manning Publications Co. Please post comments or corrections to the Author Online forum:

<http://www.manning-sandbox.com/forum.jspa?forumID=XYZ>

2. In the same table, the error code for custom validators should be `validator.invalid` not `validator.error`

## Chapter 5

### Page 139: Missing params test around register() action

The register actual should check for parameters before performing any updates (to cater for the first hit on the register action). In the updated sample below, a test for `if(params)` is added...

```
def register = {
  if (params) {
    def user = new User(params)
    if (user.validate()) {
      user.save()
      flash.message = "Successfully Create User"
      redirect(uri: '/')
    } else {
      flash.message = "Error Registering User"
      return [ user: user ]
    }
  }
}
```

### Page 146: Section 5.5.2 Uploading to the filesystem (MultipartHttpServletRequest)

The example shows the incoming file upload saved to disk as a gif even though the input format is never determined. Some users have thought that `transferTo()` did some magic transformation in the process, but this is not the case. A production implementation of this example would need to do some work on the `byte[]` and save in an appropriate format.

The text also mentions that `MultipartHttpServletRequest` is a class, whilst technically it's an interface. It's the incoming request object that implements this interface in some concrete type. Calling `getFile()` on an implementation of `MultipartHttpServletRequest` hands you back an implementation of `MultipartFile` which you can then call `transferTo()` method on. For this reason, the variable would be better named `mpf` rather than `mhr` (which is inaccurate).

Replacement text for section 5.5.2.is as follows:

### 5.5.2 Uploading to the filesystem

©Manning Publications Co. Please post comments or corrections to the Author Online forum:

<http://www.manning-sandbox.com/forum.jspa?forumID=XYZ>

### 5.5.2 Uploading to the filesystem

If you want to store the uploaded image in the filesystem rather than the database, you need access to the implementation of Spring MultipartFile that backs the upload process.

For this case, you have more options for storing the byte array:

```
def rawUpload = {  
    // a Spring MultipartFile  
    def mpf= request.getFile('photo')  
    if (!mpf?.empty && mpf.size < 1024*200) {  
        mpf.transferTo(new  
        File("/hubbub/images/${params.userId}/mugshot.gif"))  
    }  
}
```

The MultipartFile class has a transferTo() method for moving the picture data directly to a file, which is convenient if you're averse to storing BLOBs in your database. For a detailed discussion of MultipartFile, consult the Spring Javadoc.

-----  
**Note:** Line 4 of the above code segment has an annotation next to it:

**Ensures file size less than 200 Kb**

Chapter author name:

©Manning Publications Co. Please post comments or corrections to the Author Online forum:

<http://www.manning-sandbox.com/forum.jspa?forumID=XYZ>