'Hacking' directory tools

env-setup

The 'env-setup' script modifies your environment to allow you to run ansible from a git checkout using python 2.6+. (You may not use python 3 at this time).

First, set up your environment to run from the checkout:

```
$ source ./hacking/env-setup
```

You will need some basic prerequisites installed. If you do not already have them and do not wish to install them from your operating system package manager, you can install them from pip

```
$ easy_install pip               # if pip is not already available
$ pip install -r requirements.txt
```

From there, follow ansible instructions on docs.ansible.com as normal.

test-module.py

'test-module.py' is a simple program that allows module developers (or testers) to run a module outside of the ansible program, locally, on the current machine.

Example:

```
$ ./hacking/test-module.py -m lib/ansible/modules/command.py -a "echo hi"
```

This is a good way to insert a breakpoint into a module, for instance.

For more complex arguments such as the following yaml:

```yaml
parent:
  child:
    - item: first
      val: foo
    - item: second
      val: boo
```

Use:

```
$ ./hacking/test-module.py -m module \
    -a '{"parent": {"child": [{"item": "first", "val": "foo"}, {"item": "second", "val": "ba
```

return_skeleton_generator.py

return_skeleton_generator.py helps in generating the RETURNS section of a module. It takes JSON output of a module provided either as a file argument or via stdin.

fix_test_syntax.py

A script to assist in the conversion for tests using filter syntax to proper jinja test syntax. This script has been used to convert all of the Ansible integration tests to the correct format for the 2.5 release. There are a few limitations documented, and all changes made by this script should be evaluated for correctness before executing the modified playbooks.