



# Abusing pyunit To Run Regression Tests

Miki Tebeka

[mtebeka@qualcomm.com](mailto:mtebeka@qualcomm.com)



# pyunit

- `unittest` module in Python standard library
- One of the **XUnit** frameworks
- Designed for Unit Testing
- I'd like to use it for Regression Testing
  - Checks for “good” known output from a program



## Main Idea

- Have one method to run a regression
- Dynamically add test function to test class
- Expected output is in `gold` directory
- Can have input in `input` directory
- `gold/input` names are test names



# One “real” test function

```
def runtest(self, name, arguments):
    outfile = join("out", name)
    goldfile = join("gold", name)

    # Run program
    if system("echo %s > %s 2>&1" % \
              (" ".join(arguments), outfile)) != 0:
        self.fail("non-zero value return")

    # Check output
    if differ(outfile, goldfile):
        self.fail("output for %s differs" % name)
```



# Adding a Test

```
def add_test(name, arguments):  
    '''Add a test "name" with "arguments"  
    Note: "name" must be qualified Python  
    variable name  
    '''  
  
    def t(self):  
        self.runtest(name, arguments)  
  
    t.func_doc = "Testing %s" % name  
    setattr(TestEcho, "test_%s" % name, t)
```



# Adding All Tests

```
for test in glob(join("gold", "*")):  
    # Only files are tests  
    if not isfile(test):  
        continue  
  
    test = basename(test)  
    add_test(test, [test])
```



# Running

- `unittest` has a `main` function that check current source file for all class derived from `TestCase` and runs all methods starting with `test` in it

```
# Main
if __name__ == "__main__":
    main() # Imported from unittest
```



Questions?

