The Graphyte Project

design, tools and practices

Speaker: Yeukhon Wong
Michael Grossberg, Irina Gladkova,
Jeremy Neiman, Hannah Aizenman
GLASS Lab, CUNY City

April 1st, 2013
SEA Software Engineering Conference 2013
C
CARBON
12.01  2.26
Traditional Workflow
Ideal Workflow
Graphyte Architecture
Graphyte Architecture

Data Store

API Layer

Core Web Service (GCS)
Graphyte Architecture

Data Store

Version Control Host

API Layer

Core Web Service (GCS)
Graphyte Architecture

Data Store

Version Control Host

Web UI

API Layer

Core Web Service (GCS)
# Run history for climate_eof_analysis

<table>
<thead>
<tr>
<th>Start Time</th>
<th>Status</th>
<th>Run Type</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>2012-11-13 13:06:31.887932</td>
<td>Error</td>
<td>Shell Script</td>
<td>View Details</td>
</tr>
<tr>
<td>2012-11-13 14:37:49.538312</td>
<td>Error</td>
<td>Shell Script</td>
<td>View Details</td>
</tr>
<tr>
<td>2012-11-06 18:24:19.175156</td>
<td>Error</td>
<td>Shell Script</td>
<td>View Details</td>
</tr>
<tr>
<td>2012-11-06 15:07:16.260062</td>
<td>Error</td>
<td>Shell Script</td>
<td>View Details</td>
</tr>
<tr>
<td>2012-11-06 15:05:36.288337</td>
<td>Complete</td>
<td>Shell Script</td>
<td>View Details</td>
</tr>
<tr>
<td>2012-11-06 15:03:06.912129</td>
<td>Error</td>
<td>Shell Script</td>
<td>View Details</td>
</tr>
<tr>
<td>2012-10-24 12:26:45.480515</td>
<td>Error</td>
<td>Shell Script</td>
<td>View Details</td>
</tr>
</tbody>
</table>
Web UI: Result Details

- climo.000001.png
- climo.000002.png
Aurum

- A RESTful web service built using Graphyte API and infrastructure.
- Use automated tests to evaluate learning progress.
Aurum on Blackboard

**Homework Solutions**

**HW 3, Q1**
Availability: Item is no longer available. It was last available on Nov 19, 2012 11:59 PM.
Homework 3, First Question

**HW 3, Q2**
Availability: Item is no longer available. It was last available on Nov 19, 2012 11:59 PM.
Homework 3, Second Question

**HW 3, Q3**
Availability: Item is no longer available. It was last available on Nov 19, 2012 11:59 PM.
Homework 3, Third Question

**HW4 Q1**
Availability: Item is no longer available. It was last available on Dec 14, 2012 6:00 AM.
loops
List of prime numbers.

Problem: Generating Primes
In this problem you will write a Python program to generate a list of all the prime numbers between a starting number and an ending number (inclusive).

Background
A prime number is any whole number greater than 1 that is divisible only by itself and 1. Prime numbers are important in mathematics and computer science because they are used in many theorems and algorithms. Prime numbers are very commonly used in cryptography (the field of computer science devoted to keeping information secret), so there are a number of methods for generating prime numbers. Generating Primes

Problem:
1. Repeatedly ask for inputs until the start and stop numbers are greater than 0.
   - Enter 2 positive numbers:
   - 1 1
   - Enter 2 positive numbers:
   - 0 0
   - Primes between 0 and 0:

2. Print out a list of all prime numbers between the starting and stop numbers.
   - Enter 2 positive numbers: 3 7
   - Primes between 3 and 7: 3 5 7

3. Make sure it works regardless of input order.
   - Enter 2 positive numbers: 7 3
   - Primes between 3 and 7: 3 5 7

Make sure it works for arbitrary start and stop inputs:
   - Enter 2 positive numbers: 10 30
   - Primes between 10 and 30: 11 13 17 19 23 29

Tips on using this learning module:
- Do not close this window until the submit grade button at the end of the page is re-enabled.
- If you close this window during grading, you may receive a zero. Do not panic! You can always come back and submit the grade later.
- If you haven’t accepted the SSL certificate yet, please do so by right-click and open this link as new tab, accept the certificate and reload the exercise. (help video: chrome / firefox)

Enter your code:
```cpp
#include <iostream>
#include <cmath>
using namespace std;
int main(){
    int a,b, tmp;
    do{
        cout<<"Enter 2 positive numbers: ";
        cin>>a>>b;
    }while(a<0 || b<0);
    if (b<a){
        tmp = a;
        a = b;
        b = tmp;
    }
    cout<<"Primes between ",a," and ",b," ";
    int i1;
    a = 2;
    while(a<=b){
        for (i1=2;i1<=a;i1++){
```
<table>
<thead>
<tr>
<th>Test</th>
<th>Success</th>
<th>Test Case</th>
<th>Description</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Success</td>
<td>test_binary_exists</td>
<td>Does the program compile?</td>
<td>0/0</td>
</tr>
<tr>
<td>1</td>
<td>Success</td>
<td>test_primes_false</td>
<td>Correctly identifies non-prime numbers.</td>
<td>15/15</td>
</tr>
<tr>
<td>2</td>
<td>Success</td>
<td>test_primes_less_than_n</td>
<td>Prints all primes between a and b.</td>
<td>10/10</td>
</tr>
<tr>
<td>3</td>
<td>Success</td>
<td>test_primes_true</td>
<td>Correctly identifies prime numbers.</td>
<td>15/15</td>
</tr>
<tr>
<td>4</td>
<td>Success</td>
<td>test_swap_input_order</td>
<td>Makes sure the smaller number comes first.</td>
<td>5/5</td>
</tr>
<tr>
<td>5</td>
<td>Success</td>
<td>test_valid_pos_input</td>
<td>Checks that the input is positive.</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Total Score</td>
<td></td>
<td>50/50</td>
</tr>
</tbody>
</table>
In the Screen Reader mode, the table is static and grades may be entered on the Grade Details page accessed by selecting the table cell for the grade. In the interactive mode of the Grade Center, grades can be typed directly in the cells. Use the arrow keys or the tab key to navigate through the Grade Center and the Enter key to submit a grade. More Help.

<table>
<thead>
<tr>
<th>Last Name</th>
<th>First Name</th>
<th>HW 1</th>
<th>HW 2</th>
<th>Home</th>
<th>HW3 Q1</th>
<th>HW3 Q2</th>
<th>HW3 Q3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
<td>John</td>
<td>50.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Johnson</td>
<td>Jane</td>
<td>48.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Doe</td>
<td>Mary</td>
<td>0.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Brown</td>
<td>Susan</td>
<td>41.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wilson</td>
<td>Robert</td>
<td>43.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lee</td>
<td>David</td>
<td>33.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Garcia</td>
<td>Emily</td>
<td>48.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hernandez</td>
<td>Maria</td>
<td>45.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rodriguez</td>
<td>Sandra</td>
<td>15.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Perez</td>
<td>Ana</td>
<td>40.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lopez</td>
<td>Linda</td>
<td>15.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mitchell</td>
<td>Karen</td>
<td>50.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wilson</td>
<td>Emily</td>
<td>36.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Davis</td>
<td>Georgia</td>
<td>15.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wright</td>
<td>Olivia</td>
<td>20.50</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Last Saved: November 22, 2012 12:13 AM
Lessons
Good: Modular Architecture
Bad: Monolithic Implementation
Fix: Enforce modularity from beginning

The beginning is the most important part of the work.

Plato
Good:
Web service-centric
Bad:
Inconsistent API
Good:
Automated tests
Bad:
Tests take >1.5hrs
Fix: Write cheap tests

- **Unit tests**
  - (ultra fast, 100% isolation)

- **Integration tests**
  - (tolerable, b/w modules)

- **Functional tests**
  - (slow, end-to-end)
Even for prototyping, use these good practices.
Packaging Servers
Good:
Had instructions to configure machines.
Bad:
No automation, outdated Wiki
Fix:
Use automated deployment tool
Many tools, which to use?

- puppet
- CFEngine
- OPSCODE
- BCFG2
- SaltStack
What works for us

http://ansible.cc/
Ansible
Configure servers with SSH.
1. More tests, more coverage.
2. Write user and developer documentation.
4. We want VM with roles.
(database servers, web servers, cache server)
5. Open-source Graphyte!
Contributors:
Jeremy Neiman, Hannah Aizenman, Luis Bello, Yuriy Steijko, Paul K. Alabi, Joesan Gabaldon, Michael Grossberg, Irina Gladkova
Questions?

yeukhon@acm.org

glasslab.org