Google App Engine Developer Workshop

January, 2013

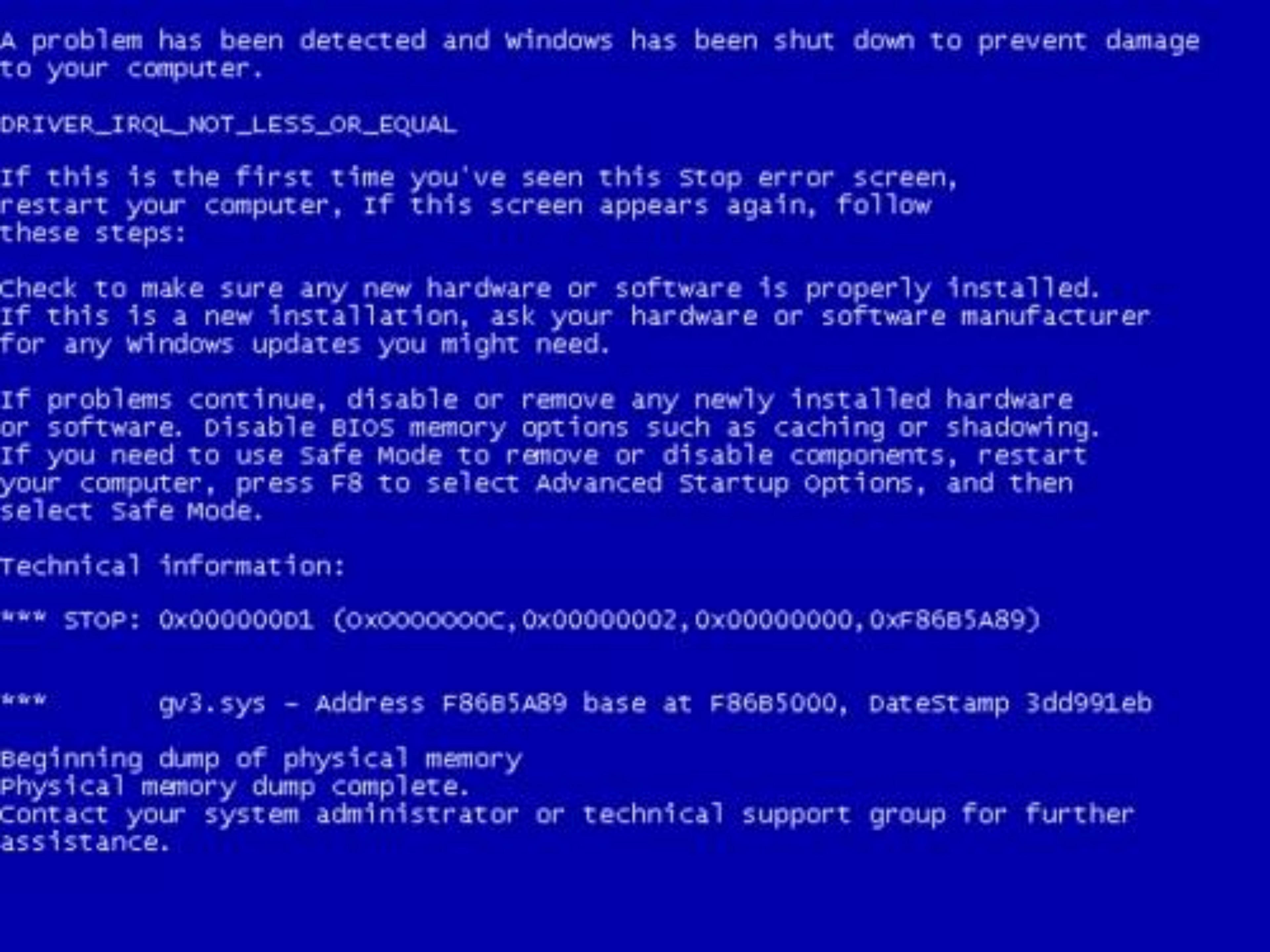
Romin Irani, Mind Storm Software Pvt. Ltd. http://www.mindstormsoftware.com

Workshop Goals

- Help you understand what is Google App Engine?
- Setup your Google App Engine account and write a Hello World App.
- Cover various App Engine APIs / Services.
- Do code lab sessions to build out an "Exam Results" App Engine application and deploy it live.
- Cover enough basics for you to take your next leap into writing Cloud Applications running on Google App Engine infrastructure.

Prerequisites

- A laptop and Internet connection
- Mobile Phone with a working phone number
- Basics of Java / JSP / Servlets



```
A problem has been detected and windows has been shut down to prevent damage
to your computer.

DRIVER_IRQL_NOT_LESS_OR_EQUAL

If this is the first time you've seen this Stop error screen,
restart your computer, If this screen appears again, follow
these steps:

Check to make sure any new hardware or software is properly installed.
If this is a new installation, ask your hardware or software manufacturer
for any Windows updates you might need.

If problems continue, disable or remove any newly installed hardware
or software. Disable BIOS memory options such as caching or shadowing.
If you need to use Safe Mode to remove or disable components, restart
your computer, press F8 to select Advanced Startup Options, and then
select Safe Mode.

Technical information:

*** STOP: 0x000000D1 (0x0000000C,0x00000002,0x00000000,0xF86B5A89)


***       gv3.sys - Address F86B5A89 base at F86B5000, DateStamp 3dd991eb

Beginning dump of physical memory
Physical memory dump complete.
Contact your system administrator or technical support group for further
assistance.
```

Session 1 – App Engine Overview

What is Google App Engine?

- Application Hosting and Development Platform
- Host your application on Google's infrastructure
- 3 runtimes supported : Java, Python and Go
- Variety of Services to boost developer productivity
- Free Quota and then Pay per use

http://cloud.google.com/appengine

App Engine – History

- April 7, 2008 : First Release (Python)
- 2009 : Support for Java
- 2011 : Go Language support
- Current Version : 1.7.4 – 13 Dec 2012
- Since 2012 - monthly release cycle

What does it provide?

- Run time infrastructure
  - Your applications run on same infrastructure that powers Google's own applications
  - Supports Applications written in Python, Java or Go
  - One Google Account – 10 Applications with Free Quota
  - Administrative Console
- Developer Tools
  - App Engine SDK
  - Local Development Server
  - Eclipse Plug-in to develop applications and 1-click deploy to Cloud

App Engine Advantages

- Leading PaaS
- Google Infrastructure
- Generous Free Quota
- Pay per use
- Quick to start : no hardware / software required from your side to host/run the application
- Global Data Locations
- 99.95 SLA
- Paid support offered as part of Premier Accounts

App Engine Services

- High Availability NoSQL and SQL Service
- Support for Java Web Applications.
  - Java version 6 recommended
- Search API
- App Engine Map Reduce
- Logs API
- SSL Support
- Pagespeed service
- Google Cloud Endpoints

App Engine Services

- Blobstore API
- Files API & integration with Google Cloud Storage (GCS)
- XMPP API
- Channel API
- Memcache API
- Users API
- Task Queues & Crons
- URL Fetch API

Usage Quotas

- Requires Google Account
- An Account can register up to 10 Applications.
- Free Quota is available for each application.
- Beyond that a pay per use policy applies

https://developers.google.com/appengine/docs/quotas

Usage – Free Quota

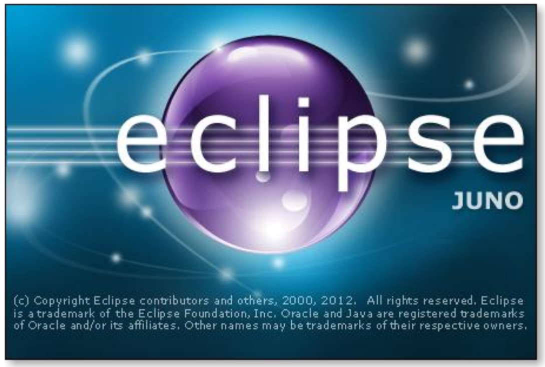
Quota	Limit (per day)
Instance-hours	28 hours
Emails	100 (5000 admin emails)
Bandwidth in	Unlimited
Bandwidth out	1 GB
Datastore	1 GB
Datastore Operations	50k
Blob Storage	5 GB
XMPP API	10k stanzas
URLFetch API calls per day	657,000

The power comes with certain restrictions like HTTP Request 60 second limit and more.

App Engine - Competition

- Amazon Web Services
- Engine Yard
- Heroku
- Force.com
- Skytap
- VMware
- Rackspace Cloud
- GoGrid
- Windows Azure
- OpenShift

Session 2 – App Engine Setup


eclipse
JUNO
(c) Copyright Eclipse contributors and others, 2000, 2012. All rights reserved. Eclipse is a trademark of the Eclipse Foundation, Inc. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

App Engine - Setup

- What you need
  - Java SDK (1.6)
  - App Engine SDK (Latest version is 1.7.4)
  - Eclipse (Take the latest version – Eclipse Juno)
  - Google Eclipse Plugin for Eclipse Juno

App Engine - Setup

- We plan to make our lives simpler.
- All the Software has been downloaded and available in a bundle, including Eclipse that is configured with the Google Eclipse Plugin.
- Go to /**hands-on-exercises** and follow **App Engine Dev Environment Setup.docx** to get your Dev Environment ready!
- ***Optional:*** *If you wish to install the App Engine environment from scratch i.e. download Java, Eclipse, Google Plugin, refer to /**hands-on-exercises** folder and refer From Scratch **From Scratch - Google App Engine Development Environment Setup.docx.***

Hands On Exercise

- Our goal here is to setup your Development Environment.
- Go to /**hands-on-exercises** folder.
- Follow **App Engine Dev Environment Setup.docx**

Session 3 – Hello World

App Engine – Hello World

- Get our App Engine Account Setup and create an Application.
- Develop the default App Engine application using Google Eclipse Plugin Project Wizard.
- Deploy our Application into the App Engine cloud
- See it live !
- While the application does not do much, it will demonstrate your development / deploy workflow.

App Engine – Account Setup

- Go to http://appengine.google.com
- Sign in with your Google Account.
- When you create Applications for the first time, it will ask for Mobile Number and Google will send you a verification code via SMS.
- Complete the validation process with the code.
- Each Google Account gives you 10 Applications
- A Mobile number can be verified only once ;-)

App Engine – Create App

- We do not want to stray from tradition, so let us create a **Hello World** application.
- Use Eclipse and Google Eclipse plugin to generate default Application Template.
- Test it out locally.

Mind Storm

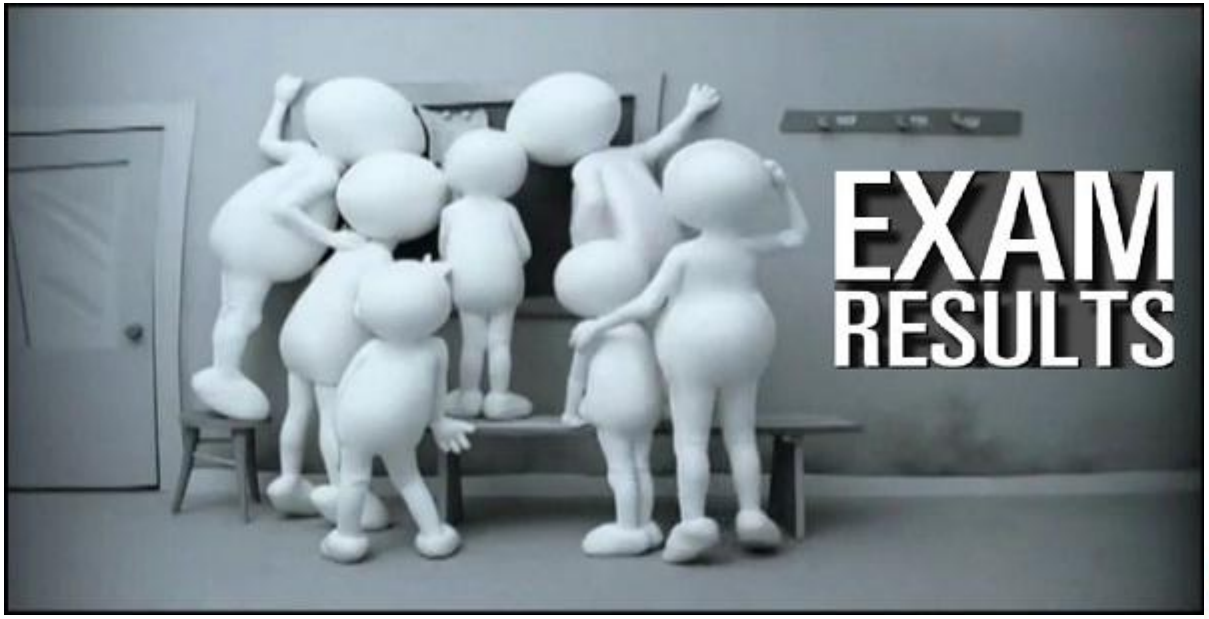
App Engine – Deploy App

- Final Step is to deploy the App to the Cloud.
- Step 1 : Create an Application that will host your application. This is a unique Application Id. This is done from http://appengine.google.com. Login and go to My **Applications** and then **Create Application.**
- Step 2 : Use the Google one-click Deploy option in Eclipse to deploy your application directly under that Application Id.
- The app on successful deployment will be available at **http://<APPID>.appspot.com**

Hands On Exercise

- Go to **/hands-on-exercises/HelloWorld.docx**
- Follow the document and deploy your First Google App Engine application.

Session 4 – Exam Results App



ExamResults App

- Web Application
- Lookup Exam Results
- Web Interface to enter your Seat Number and it gives back results.
- Accepts request over Google Chat and responds with Chat message.
- Accepts request over Email and responds with Email.

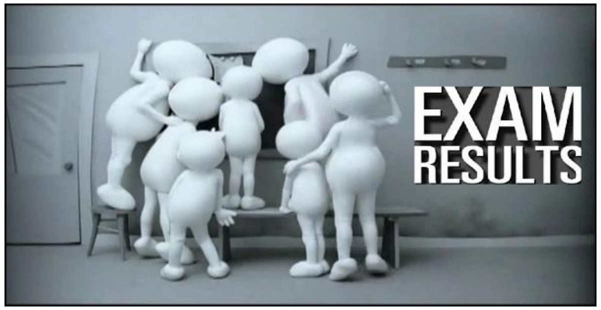
Demos

- Exam Results App – In Action

http://exam-results.appspot.com

Session 5 – Code Lab

Web Interface



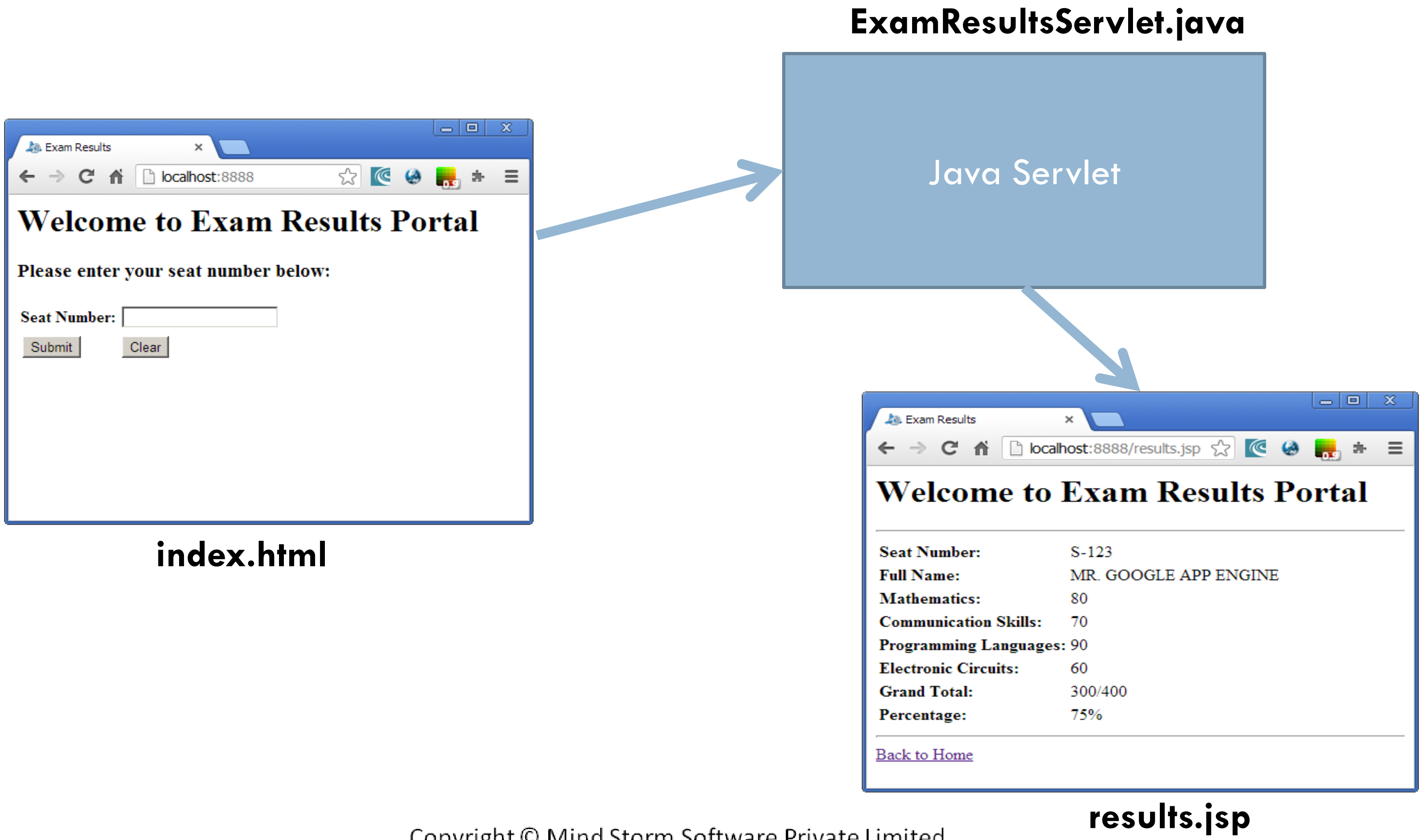
Web Interface

- We will start off with building a Web Interface as demonstrated in the demo
- Make use of Servlets and JSP
- App Engine supports Java Servlet 2.5 specification
- If you are familiar with developing basic Java Servlet / JSP applications, you are all set.

Web Interface – The Flow

- The flow is simple:
  - Home page (index.html) that has a form that accepts a Seat Number of the student who took the exam
  - This is submitted to a Servlet, which builds a dummy result
  - The Servlet forwards control to a JSP page that will display the Exam result
  - If there are any errors, an error JSP page that is configured will display the error

Web Interface – Visual Flow



Exam Results

localhost:8888

Welcome to Exam Results Portal

Please enter your seat number below:

Seat Number:

Submit Clear

Exam Results

localhost:8888/results.jsp

Welcome to Exam Results Portal

Seat Number:	S-123
Full Name:	MR. GOOGLE APP ENGINE
Mathematics:	80
Communication Skills:	70
Programming Languages:	90
Electronic Circuits:	60
Grand Total:	300/400
Percentage:	75%

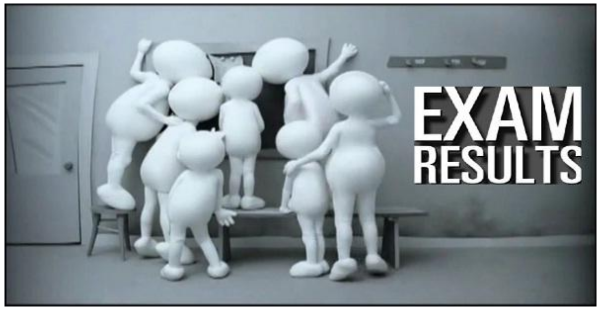
Back to Home

Hands On Exercise

- Step by Step instructions to create the Exam Results Application and the Web Interface. It contains:
  - Create a default Template for the Exam Results Application. Go to **/hands-on-exercises/ExamResults-Step1.docx**
  - Build on that by developing the Web Interface. Go to **/hands-on-exercises/ExamResults-Step2-WebInterface.docx**

Session 6 – Code Lab

Datastore Service



App Engine Datastore

- App Engine provides a highly scalable option for persisting your data
- It is called the Datastore. The Datastore API is used to persist and retrieve data.
- The Datastore API uses Google BigTable.
- The datastore supports two standard Java interfaces: Java Data Objects (JDO) 2.3 and Java Persistence API (JPA) 1.0
- Various 3rd party libraries also exist to make life easier. E.g. Objectify

App Engine Datastore

- We shall use JDO Annotations to setup an Entity class to be persistent ready

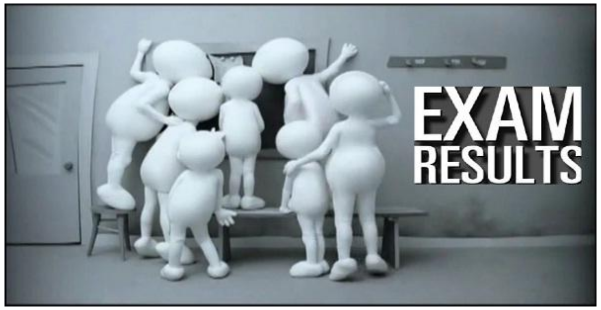
```
@PersistenceCapable
public class ExamResult implements Serializable {

    @PrimaryKey
    @Persistent(valueStrategy = IdGeneratorStrategy.IDENTITY)
    Key     seatNumber;
    @Persistent
    String studentName;
    @Persistent
    String marks_Math;
    ...
}
```

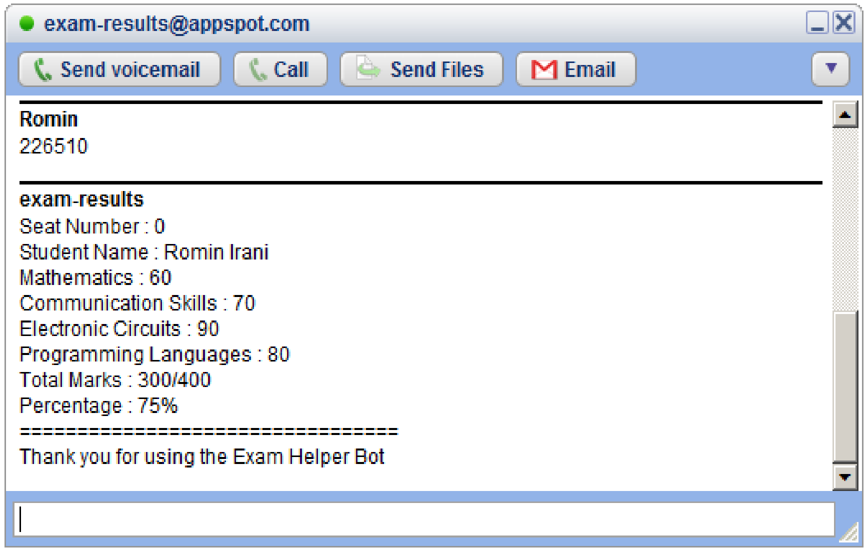
Hands On Exercise

- Step by Step instructions to create the Datastore Layer for the application.
- We will build a persistable Entity named ExamResult and use the JDO API to write and search for the records. It will our Data Access Object (DAO).
- Additionally, we will integrate it with the ExamResultsServlet code so that actual data is returned.
- Go to /**hands-on-exercises**/ **ExamResults-Step3-Datastore.docx**

Session 7 – Code Lab

XMPP Service





```
Romin
226510

exam-results
Seat Number : 0
Student Name : Romin Irani
Mathematics : 60
Communication Skills : 70
Electronic Circuits : 90
Programming Languages : 80
Total Marks : 300/400
Percentage : 75%
==================================
Thank you for using the Exam Helper Bot
```

App Engine – XMPP Service

- Instant Messages or Chat.
- App Engine Application can talk to any XMPP-compatible Chat Service such as Google Talk or Jabber.
- Useful for creating interactive Chat based applications that a user can add to Google Talk and communicate via a subset of commands.
- For e.g. A Weather XMPP Bot which when provided a city name, gives its current weather conditions.

App Engine – XMPP Service

- XMPP Service support includes
  - Sending XMPP Messages
  - Receiving XMPP Messages
  - Sending Invitations
  - Managing Presence

https://developers.google.com/appengine/docs/java/xmpp/

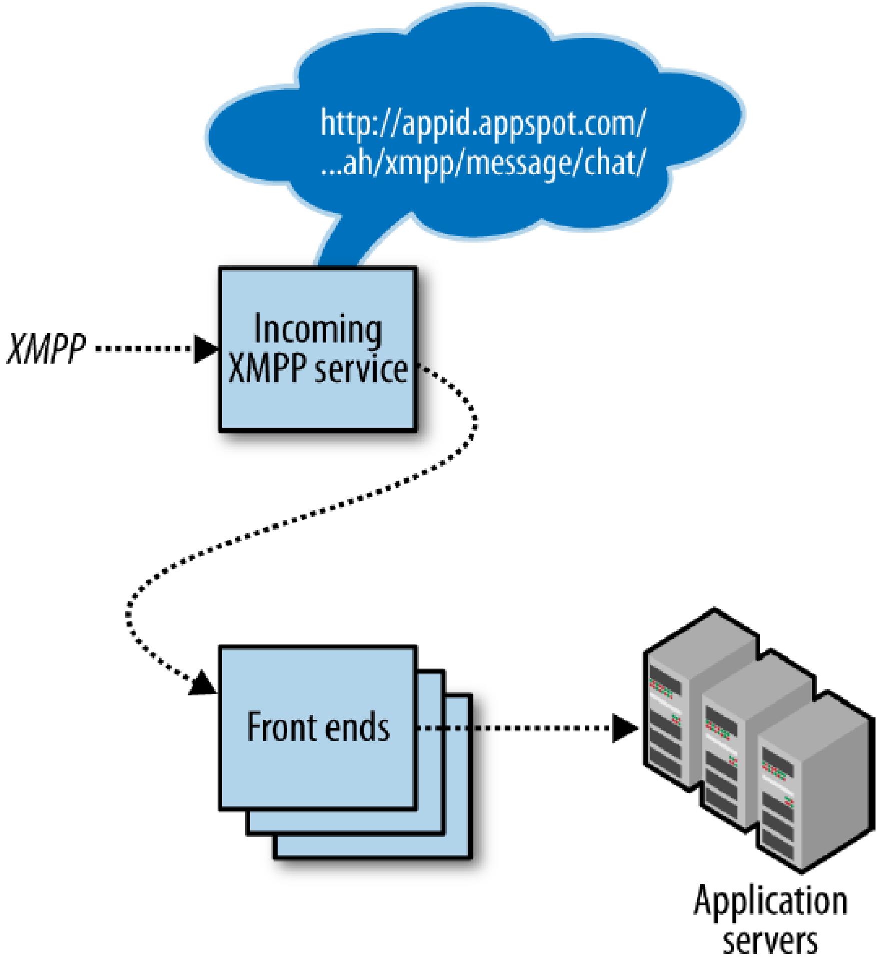
App Engine – XMPP Service

- Sending XMPP message is as simple as invoking the similar to invoking any HTTP request, simply invoke the XMPP API to send the message. Optionally, you can monitor for errors.
- Receiving XMPP is similar to receiving HTTP requests. App Engine wraps the message and invokes standard Request Handlers for XMPP in your application.
- The message is put in the HTTP Request payload and you can parse it out.

App Engine – XMPP JID

- Each participant in an XMPP Application is identified by a Jabber ID (JID)
- Jabber ID : username @ domain / resource
- To send a XMPP Message, you send the message to your own server, which delivers it to the other server and if the user is online, the message is delivered to the client application e.g. Google Talk.
- For XMPP Chat, your App Engine application can receive messages at app-id@appspot.com or anything@app-id.appspotchat.com

App Engine – XMPP Service



App Engine – Receiving XMPP Messages

- To receive XMPP Message in your application, first enable the feature in the application configuration.
- Go to **appengine-web.xml** in /**WEB-INF** folder and add the following entry:

```
<inbound-services>
        <service> xmpp_message </service>
</inbound-services>
```

App Engine – Receiving XMPP Messages

- Add the incoming XMPP Message URL Path to a Servlet
- **web.xml** description

```
<servlet>
  <servlet-name>xmppreceiver</servlet-name>
  <servlet-class>myapp.XMPPReceiverServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>xmppreceiver</servlet-name>
  <url-pattern>/_ah/xmpp/message/chat/</url-pattern>
</servlet-mapping>
```

- In the Servlet, handle the POST method to extract out the XMPP message payload

App Engine – Receiving XMPP Message

```
public class XMPPReceiverServlet extends HttpServlet {
public void doPost(HttpServletRequest req, HttpServletResponse resp)
throws IOException {

XMPPService xmpp = XMPPServiceFactory.getXMPPService();
Message message = xmpp.parseMessage(req);
// ... Use methods like
// message.getFromJid();
// message.getBody();

}

}
```

App Engine – Sending XMPP Message

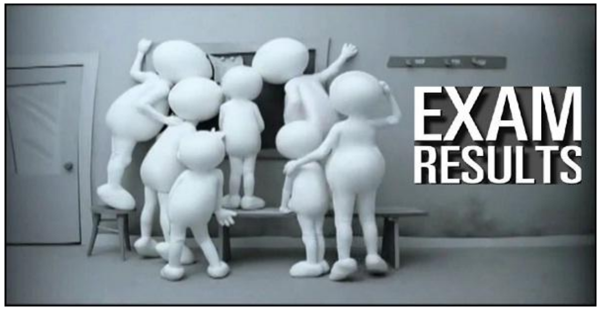
```
XMPPService xmpp = XMPPServiceFactory.getXMPPService();

JID recipient = new JID(“romin.k.irani@gmail.com");

Message message = new MessageBuilder()
.withRecipientJids(recipient)
.withBody(“Welcome to App Engine!")
.build();

SendResponse success = xmpp.sendMessage(message);
if (success.getStatusMap().get(recipient) !=endResponse.Status.SUCCESS) {
// ...
}
```

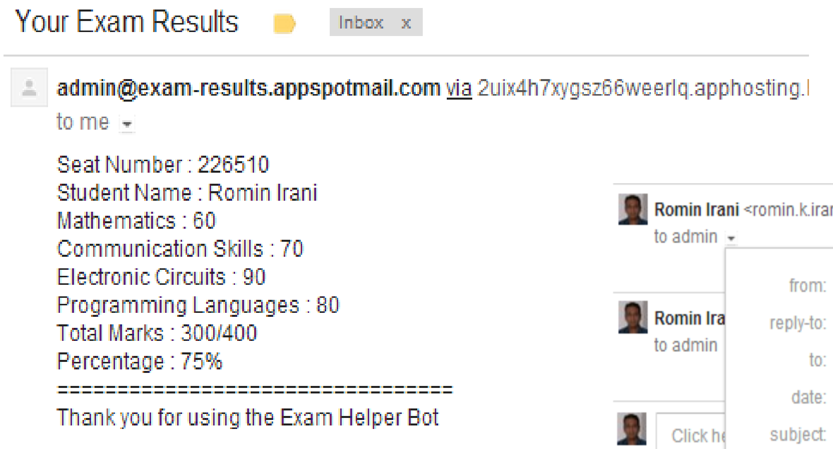
Hands On Exercise

- Allow for anyone to request their exam result via XMPP Chat
- Write incoming XMPP handler to receive and parse request
- Send out the response via XMPP itself
- Flow is: User logs to Google Talk, adds the Application Chat Id to his/her friends list and interacts with the XMPP Chat Application
- Go to **/hands-on-exercises/ExamResults-Step4-XMPP.docx**

Session 8 – Code Lab

Email Service

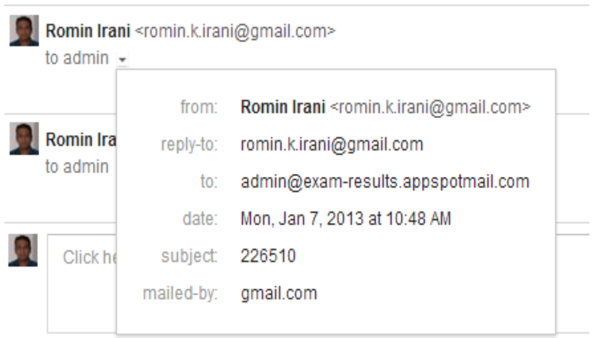



Your Exam Results Inbox x

admin@exam-results.appspotmail.com via 2uix4h7xygsz66weerlq.apphosting.

to me

Seat Number : 226510
Student Name : Romin Irani
Mathematics : 60
Communication Skills : 70
Electronic Circuits : 90
Programming Languages : 80
Total Marks : 300/400
Percentage : 75%
==================================
Thank you for using the Exam Helper Bot


Romin Irani <romin.k.irani@gmail.com>
to admin

from:	Romin Irani <romin.k.irani@gmail.com>
reply-to:	romin.k.irani@gmail.com
to:	admin@exam-results.appspotmail.com
date:	Mon, Jan 7, 2013 at 10:48 AM
subject:	226510
mailed-by:	gmail.com

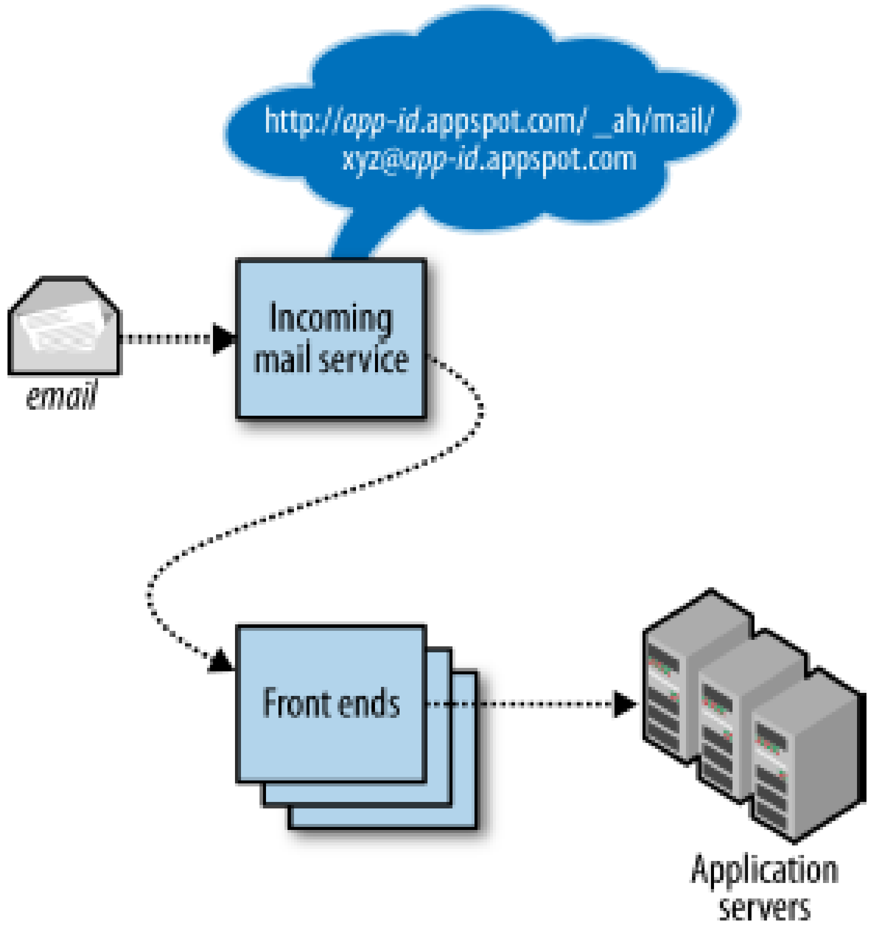
App Engine – Mail Service

- App Engine provides the Mail Service API
- Used for sending and receiving email
- Sending Email is similar to invoking any HTTP request, simply invoke the Mail API to send email.
- Receiving Email is similar to receiving HTTP requests. App Engine wraps the message and invokes standard Request Handlers for Mail in your application
- The message is put in the HTTP Request payload
- https://developers.google.com/appengine/docs/java/mail/

App Engine – Mail Service

- Each app has its own set of incoming email addresses, based on its application ID.
- For email, the app can receive messages at addresses of these forms:
  - *app-id@appspotmail.com*
  - *anything@app-id.appspotmail.com*
- Free Quota : 100 per day external, 5000 per day (Administrator)
- Dev Server supports API but does not send out actual email. It just shows it in the Log.

App Engine – Mail Service



App Engine – Receiving Email

- To receive email in your application, first enabled the feature in the application configuration
- Go to **appengine-web.xml** in /**WEB-INF** folder and add the following entry:

```
<inbound-services>
        <service>mail</service>
</inbound-services>
```

App Engine – Receiving Email

- Add the incoming email URL Path to a Servlet
- **web.xml** description

```
<servlet>
  <servlet-name>mailreceiver</servlet-name>
  <servlet-class>myapp.MailReceiverServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>mailreceiver</servlet-name>
  <url-pattern>/_ah/mail/*</url-pattern>
</servlet-mapping>
```

- In the Servlet, handle the POST method to extract out the Email message payload

App Engine – Receiving Email

```
public void doPost(HttpServletRequest req, HttpServletResponse resp) throws
IOException {
        Properties props = new Properties();
        Session session = Session.getDefaultInstance(props, null);
        try {
                MimeMessage message = new MimeMessage(session, req.getInputStream());
                String contentType = message.getContentType();
                Object content = message.getContent();
                if (content instanceof String) {
                // A plain text body.
                }
                else if (content instanceof Multipart) {}
        }
        catch (MessagingException e) {
        //..
        }
}
```

App Engine – Sending Email

- To send Email, call the API of the Mail Service
- Fill out From, To, CC, BCC, Subject, Message, Attachments.
- Valid From Address
  - The Google Account address of one of the application administrators
  - The address of the user currently signed in to the app
  - A valid incoming email address for the application

App Engine – Sending Email

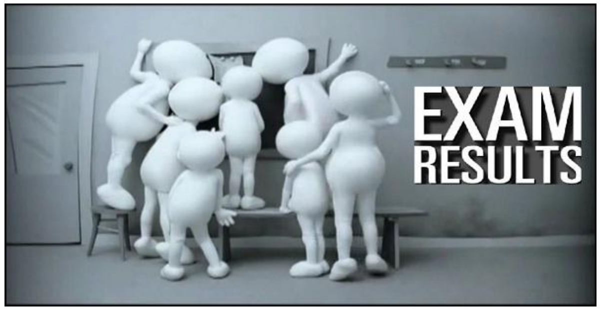
```
String recipientAddress = <email address>;
Properties props = new Properties();
Session session = Session.getDefaultInstance(props, null);
String messageBody =“Hi”;
try {
      Message message = new MimeMessage(session);
      message.setFrom(new InternetAddress(“test@test.com"));
      message.addRecipient(Message.RecipientType.TO, new
      InternetAddress(recipientAddress));
      message.setSubject("Welcome to App Engine Workshop!");
      message.setText(messageBody);
      Transport.send(message);
}
catch (Exception e) {
}
```

Hands On Exercise

- Allow for anyone to request their exam result via Email
- Write incoming email handler to receive and parse request
- Send out the response via email (Ideally this should be send out asynchronously via Task Queue API)
- Go to /**hands-on-exercises/ExamResults-Step5-Email.docx**

Session 9 – Code Lab

Cron Service



App Engine – Cron Service

- Ability to run tasks at regular intervals
- Schedule one or more repetitive tasks to run at different intervals
- Ability to specify Time Zones also
- Examples
  - Fetch the latest news every 1 hour
  - Update App statistics every 6 hours
  - Check system to send any email reminders every 12 hours
  - Take a backup every week on Wednesday at 12:00 AM
- https://developers.google.com/appengine/docs/java/config/cron

App Engine – Cron Service

- Cron Schedules are specified in a **cron.xml** file that is placed in the /**WEB-INF** directory

```
<cronentries>
  <cron>
    <url>/cron/reports</url>
    <description>Send nightly reports.</description>
    <schedule>every day 23:59</schedule>
    <timezone>America/Los_Angeles</timezone>
  </cron>
  <cron>
    <url>/cron/getnews</url>
    <description>Refresh news.</description>
    <schedule>every 1 hours</schedule>
  </cron>
</cronentries>
```

App Engine – Cron Service

- Each Cron Entry specifies the following:
  - **description** : A description of the task
  - **url** : the URL path of the Request handler to invoke. This will map to the Servlet that you will configure in the **web.xml** file
  - **schedule:** the Cron expression that specifies the schedule on which to execute the task
  - **timezone:** Optional. The Zone Info descriptor for the timezone e.g. America/Los Angeles. If omitted it is UTC
  - **target:** Optional. ID of the App version to use for the task.

Cron Service - Schedules

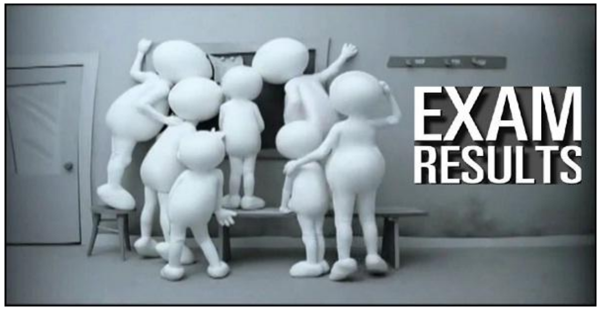
- A Cron Expression for the schedule is a simplified English-like format that describes the recurrence of the task
- every 1 minutes
- every 10 hours
- every day 11:00
- every wednesday 15:00
- 2nd sunday
- every 30 mins from 9:00 to 12:00

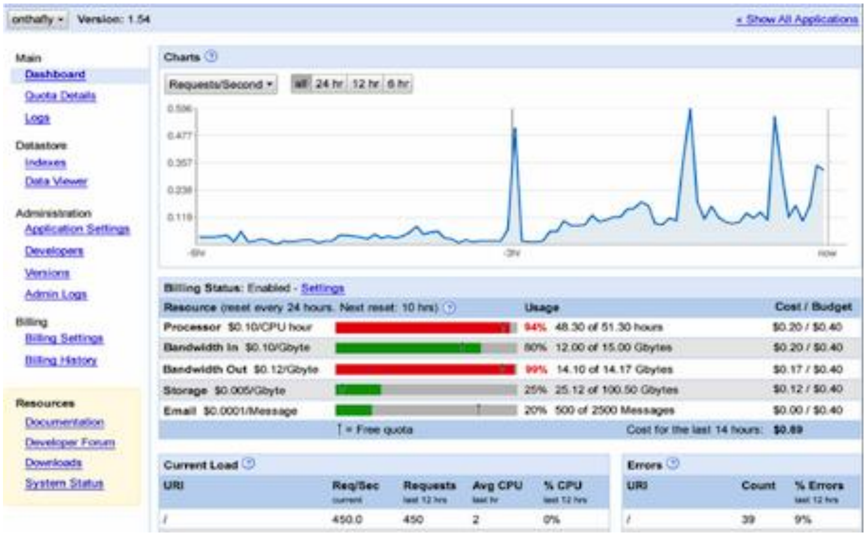
Hands On Exercise

- Write out a Cron Job that executes every 2 or 1 hour and prints out a dummy string.
- First configure your cron.xml file with the cron entry and place it in /**WEB-INF** folder
- Map the Cron Job Request Handler to Servlet and code the Servlet
- Deploy the application on App Engine and observe the results
- Go to /**hands-on-exercises/ExamResults-Step6-CronJob.docx**

Session 10 – Code Lab

Administrative Console





Billing Status: Enabled - Settings

Resource (reset every 24 hours. Next reset: 10 hrs)		Usage	Cost / Budget
Processor $0.10/CPU hour	94%	48.30 of 51.30 hours	$0.20 / $0.40
Bandwidth In $0.10/Gbyte	80%	12.00 of 15.00 Gbytes	$0.20 / $0.40
Bandwidth Out $0.12/Gbyte	99%	14.10 of 14.17 Gbytes	$0.17 / $0.40
Storage $0.005/Gbyte	25%	25.12 of 100.50 Gbytes	$0.12 / $0.40
Email $0.0001/Message	20%	500 of 2500 Messages	$0.00 / $0.40
↑ = Free quota		Cost for the last 14 hours:	$0.89

Current Load

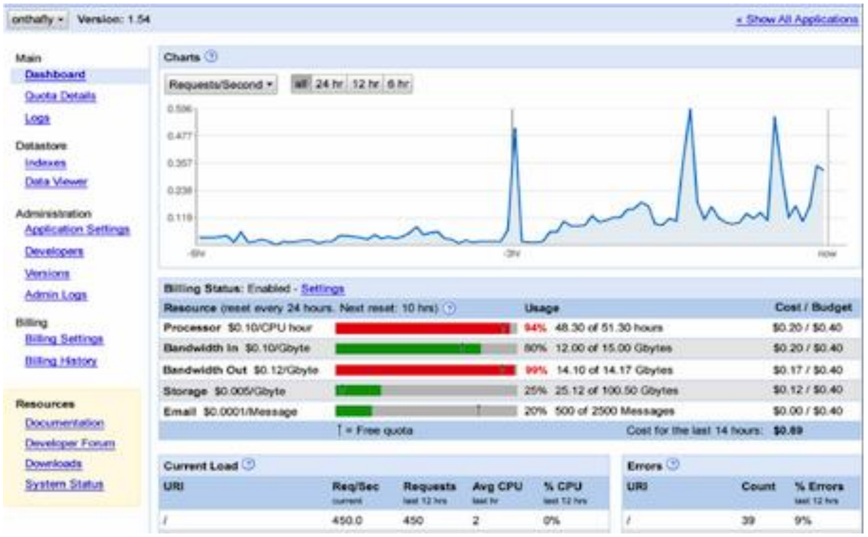
URI	Req/Sec current	Requests last 12 hrs	Avg CPU last hr	% CPU last 12 hrs
/	450.0	450	2	0%

Errors

URI	Count	% Errors last 12 hrs
/	39	9%

Administrative Console

- Login at http://appengine.google.com
- Visit your list of applications
- Click on a particular application to view the Administrative Console



Billing Status: Enabled - Settings

Resource (reset every 24 hours. Next reset: 10 hrs)	Usage		Cost / Budget
Processor $0.10/CPU hour	94%	48.30 of 51.30 hours	$0.20 / $0.40
Bandwidth In $0.10/Gbyte	80%	12.00 of 15.00 Gbytes	$0.20 / $0.40
Bandwidth Out $0.12/Gbyte	99%	14.10 of 14.17 Gbytes	$0.17 / $0.40
Storage $0.005/Gbyte	25%	25.12 of 100.50 Gbytes	$0.12 / $0.40
Email $0.0001/Message	20%	500 of 2500 Messages	$0.00 / $0.40

↑ = Free quota Cost for the last 14 hours: **$0.69**

Current Load

URI	Req/Sec current	Requests last 12 hrs	Avg CPU last hr	% CPU last 12 hrs
/	450.0	450	2	0%

Errors

URI	Count	% Errors last 12 hrs
/	39	9%

Administrative Console

- Dashboard shows current usage / quota limits
- View Logs
- View Datastore
- Versions
- Enable/Disable Application
- and more …

Hands On Exercise

- For one of your App Engine applications, login with your account credentials at http://appengine.google.com
- Visit the Dashboard and visit the different links
- Most important ones:
  - Current usage
  - Logs
  - Datastore viewer

More Services

- We have only seen a few of the App Engine Services
- Investigate and learn about these additional ones
  - Blobstore – for storing large amounts of data
  - Memory Cache – store and retrieve data from memory
  - Task Queues – Create asynchronous tasks that are executed by App Engine automatically
  - And more …
  - Visit : https://developers.google.com/appengine/docs/java/

Next Steps

- Take a break
- App Engine application ideas
- Put down on paper what you want your app to do
- Break down your app into App Engine services needed.

Now for some ideas ….

App Engine Ideas – List 1

- Quiz
- Monitor Social Media for mentions
- Customer Support System / Tickets – driven via Email
- Medical Applications
  - Track activities / food
  - Monitor Health Incidents
  - Vaccination Reminders
  - Alternative Drugs

App Engine Ideas – List 2

- Game Scoring Platform
  - API driven, Game -> High Scores, Leaderboard, etc
- Speaker Feedback System
- Employee / Kids Tracking System
- Ad Platform
- Document Sharing , Photo Sharing
- Exam Results
- Asset Management

App Engine Ideas – List 3

- XMPP Bots
  - Campus Information
  - Events
  - Local Goa stuff
  - Train / Bus / Flight
- Civic Apps
  - Accident / Traffic Reporting
  - Neighborhood Issues

App Engine Applications

- App Engine applications can drive any kind of client
- It can be :
  - Web Browser
  - Mobile Application (Android, iPhone)
  - Chrome Browser Extension
- Get ! Set ! Go !
- All The Best !

Important URLs

- Official Site : https://cloud.google.com/products/
- AppEngine Console: http://appengine.google.com
- Java Documentation: https://developers.google.com/appengine/docs/java/overview
- AppEngine Code Lab: http://googcloudlabs.appspot.com

Books

- Excellent in-depth book on Google App Engine covering both Python and Java
- Code snippets in this presentation taken from book for demonstration purpose only

Programming Google App Engine, 2nd Edition

By Dan Sanderson
Publisher: O'Reilly Media
Released: October 2012
Pages: 538

http://shop.oreilly.com/product/0636920017547.do

Thank You

- Q & A
- Website : http://www.mindstormsoftware.com
- Email : romin.irani@mindstormsoftware.com
- Blog : http://www.romin.irani.com
- App Engine Tutorials : http://www.rominirani.com/category/cloud-computing/google-app-engine/
- Free Book on App Engine (dated but things should work with minor syntax changes in some examples): http://www.rominirani.com/2010/03/09/gaej-experiments-ebook/

Appendix

- Some extra slides here
- An Important one is on Java restrictions.
- At the end of the day, App Engine is a PaaS and it puts some restrictions on what you can and more importantly cannot do. So certain Java classes might not be supported, etc.

App Engine – Do Note

- Current supported languages : Python, Java & Go
- It supports various JVM supported languages like Groovy, JRuby, Scala, Clojure, etc.
- Supports standard Java Servlet 2.5 technology using Jetty Open Source Web Server
- Several frameworks like Spring, Struts are supported with workarounds
- Some APIs are experimental in nature. They could undergo changes till they reach Production status

App Engine – Java Restrictions

- Developers have read-only access to the filesystem on App Engine.
- App Engine can only execute code called from an HTTP request (scheduled background tasks allow for self calling HTTP requests).
- Java applications may only use a subset (The JRE Class White List) of the classes from the JRE standard edition.
- Datastore cannot use inequality filters on more than one entity property per query.
- A process started on the server to answer a request can't last more than 60 seconds (with the 1.4.0 release, this restriction does not apply to background jobs anymore).

App Engine – XMPP Errors

- When you invoke the XMPP Service to sending a message, the message is queued for delivery and the only error you will get is if the message is malformed
- XMPP Service puts all incoming error messages onto a separate inbound service called **xmpp_error**
- To enable this service, add the **xmpp_error** inbound service to the application configuration in **appengine-web.xml**

AppEngine – XMPP Errors

- The entry to add to **appengine-web.xml** is shown below:

```
<inbound-services>
<service>xmpp_message</service>
<service>xmpp_error</service>
</inbound-services>
```

- Error messages arrive as POST requests at this URL path:/**_ah/xmpp/error/**
- You need to map the Request Handler i.e. a Servlet to the above path to receive the error messages.