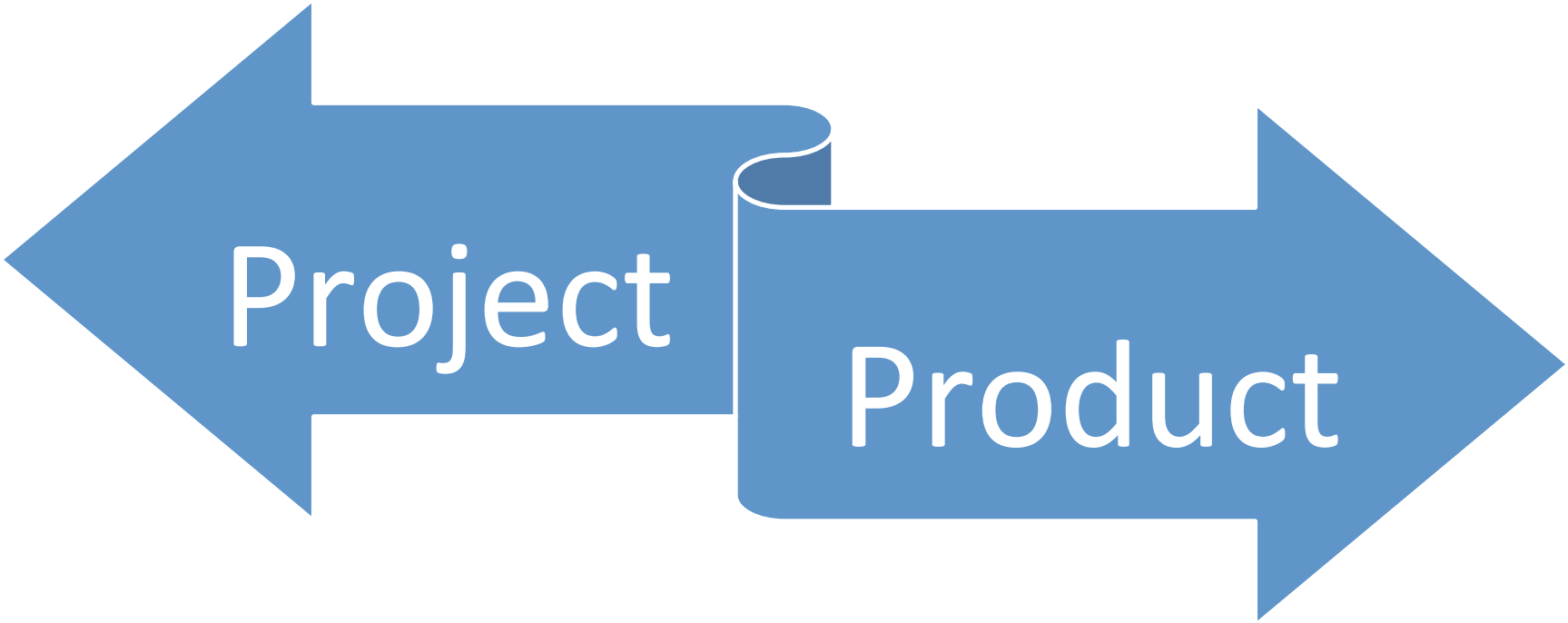


Thanks to Nilesh Bansal for material in these slides

REAL WORLD

Core Difference

- **Project**
  - You define the problem statement

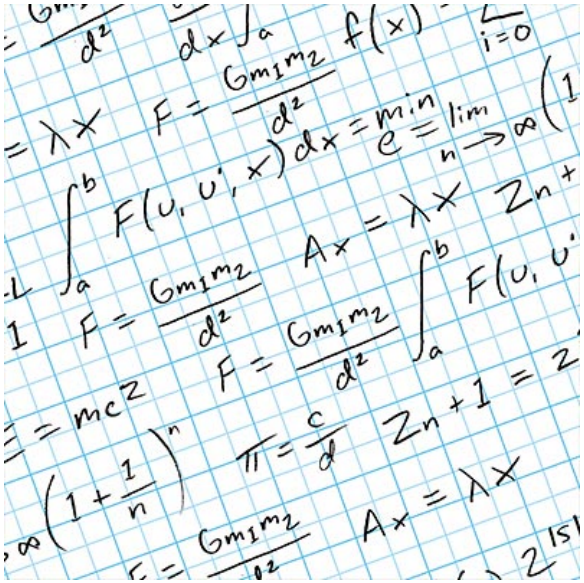
Core Difference

- **Project**
  - You define the problem statement
  - You are the primary user

INPUT DATE: 03 / 06 / 2010

Core Difference

- **Project**
  - You define the problem statement
  - You are the primary user
  - Often technically challenging and novel



Core Difference

- **Project**
  - You define the problem statement
  - You are the primary user
  - Often technically challenging and novel

- **Product**
  - Solving someone else's problem
  - Someone else is using the product
  - Often **simple solutions** to complex problems

Overview

- Usability
- Security
- Reliability
- Performance

USABILITY

Usability

- **Most important, but often overlooked**
  - Very difficult task

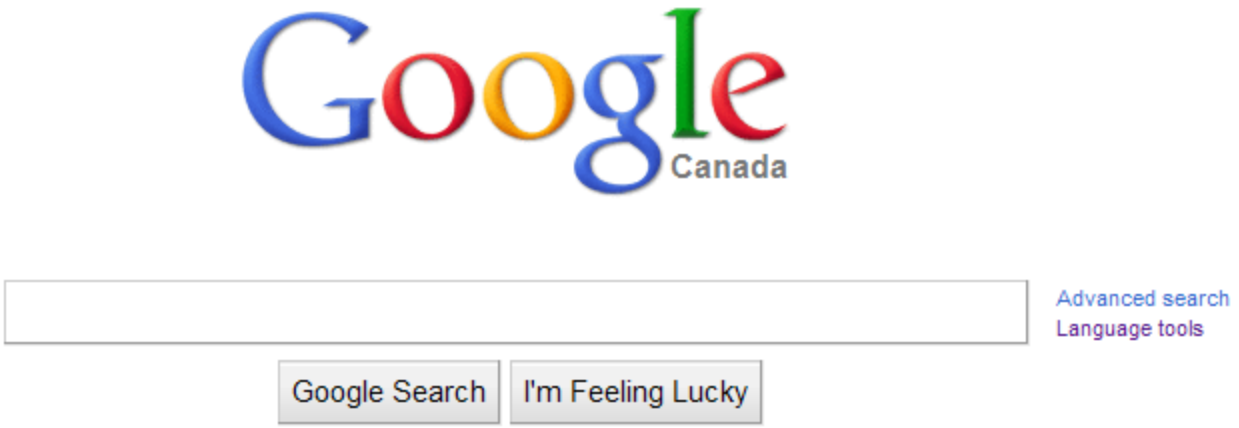
Usability

- **Most important, but often overlooked**
  - Very difficult task
  - Silently observe user

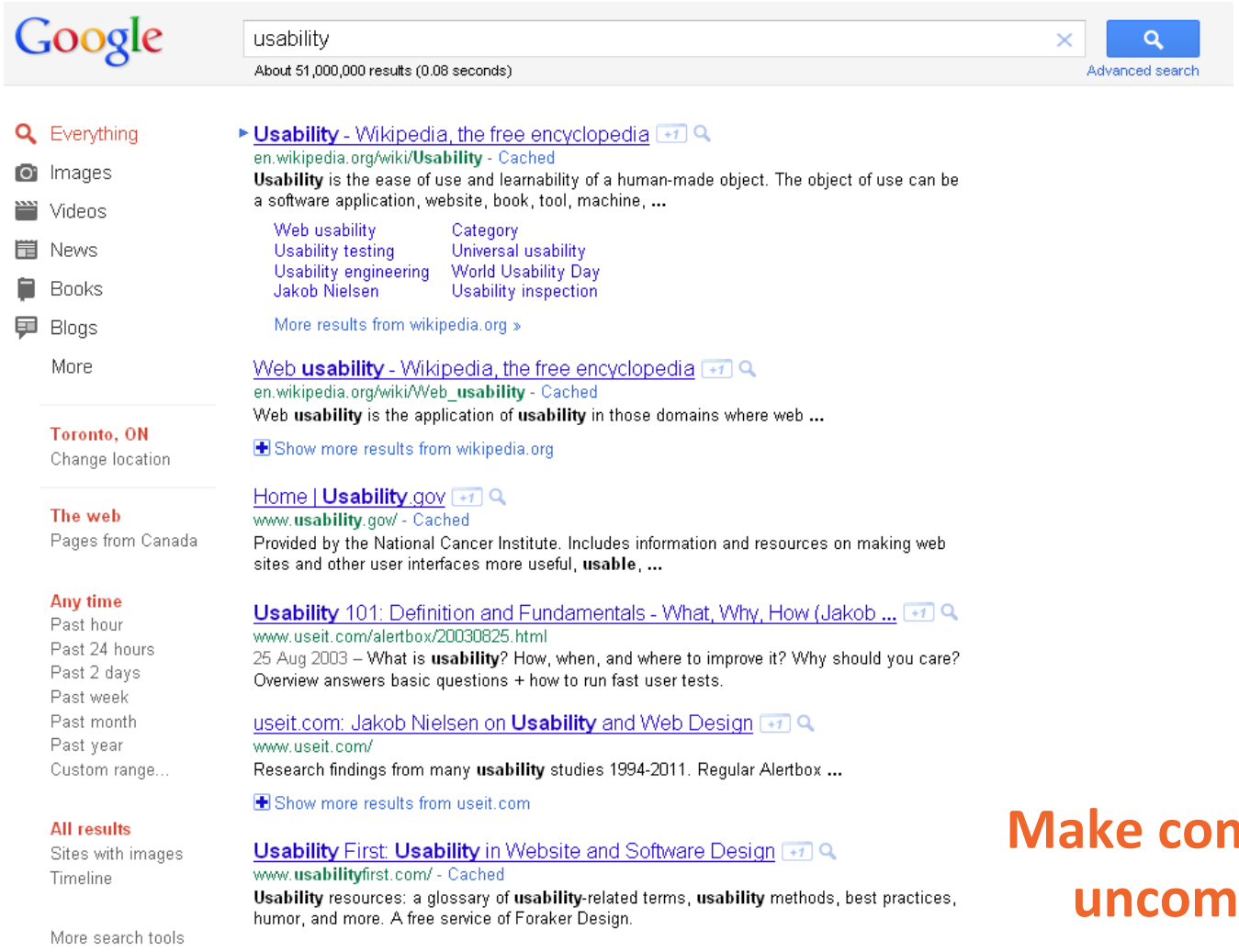
Make common easy,

and uncommon possible

Google Search



Google Search



Make common easy, uncommon possible

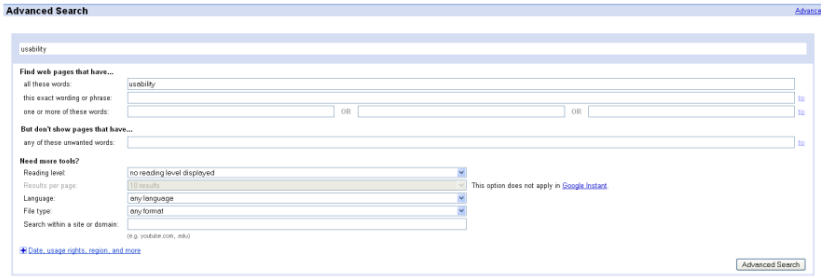
Google Search

Google

usability

About 51,000,000 results (0.08 seconds)

Advanced search



- Everything
- Images
- Videos
- News
- Books
- Blogs
- More

Toronto, ON
Change location

The web
Pages from Canada

Any time
Past hour
Past 24 hours
Past 2 days
Past week
Past month
Past year
Custom range...

All results
Sites with images
Timeline

More search tools

Usability - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/**Usability** - Cached
Usability is the ease of use and learnability of a human-made object. The objec
a software application, website, book, tool, machine, ...

Web usability
Usability testing
Usability engineering
Jakob Nielsen
Category
Universal usability
World Usability Day
Usability inspection

More results from wikipedia.org »

Web **usability** - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/Web_**usability** - Cached
Web **usability** is the application of **usability** in those domains where web ...

Show more results from wikipedia.org

Home | **Usability**.gov
www.**usability**.gov/ - Cached
Provided by the National Cancer Institute. Includes information and resources on making web sites and other user interfaces more useful, **usable**, ...

Usability 101: Definition and Fundamentals - What, Why, How (Jakob ...
www.useit.com/alertbox/20030825.html
25 Aug 2003 – What is **usability**? How, when, and where to improve it? Why should you care? Overview answers basic questions + how to run fast user tests.

useit.com: Jakob Nielsen on **Usability** and Web Design
www.useit.com/
Research findings from many **usability** studies 1994-2011. Regular Alertbox ...

Show more results from useit.com

Usability First: **Usability** in Website and Software Design
www.**usability**first.com/ - Cached
Usability resources: a glossary of **usability**-related terms, **usability** methods, best practices, humor, and more. A free service of Foraker Design.

Make common easy, uncommon possible

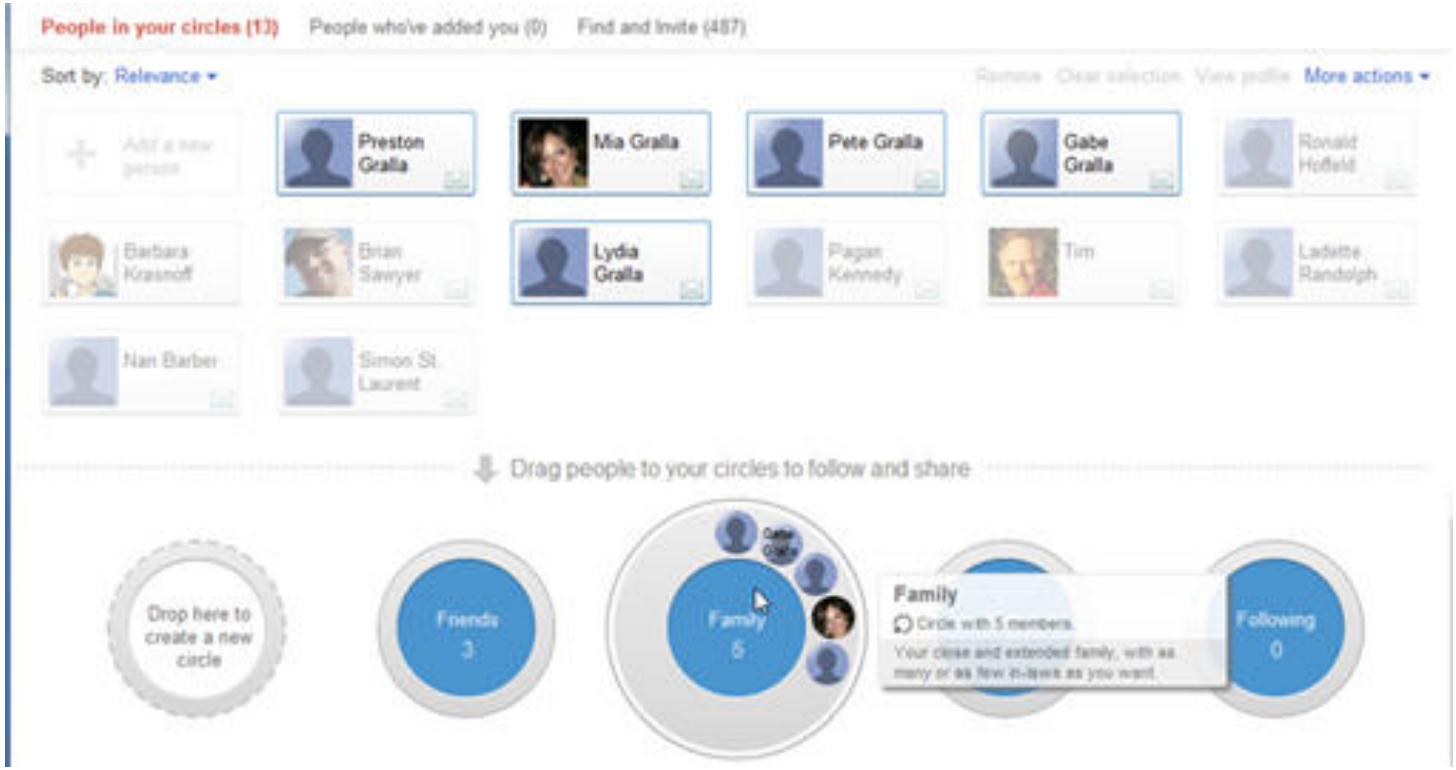
Reasonable Defaults

- **Customizable interface**
  - But 90% users do not customize
  - They use what system provides by default

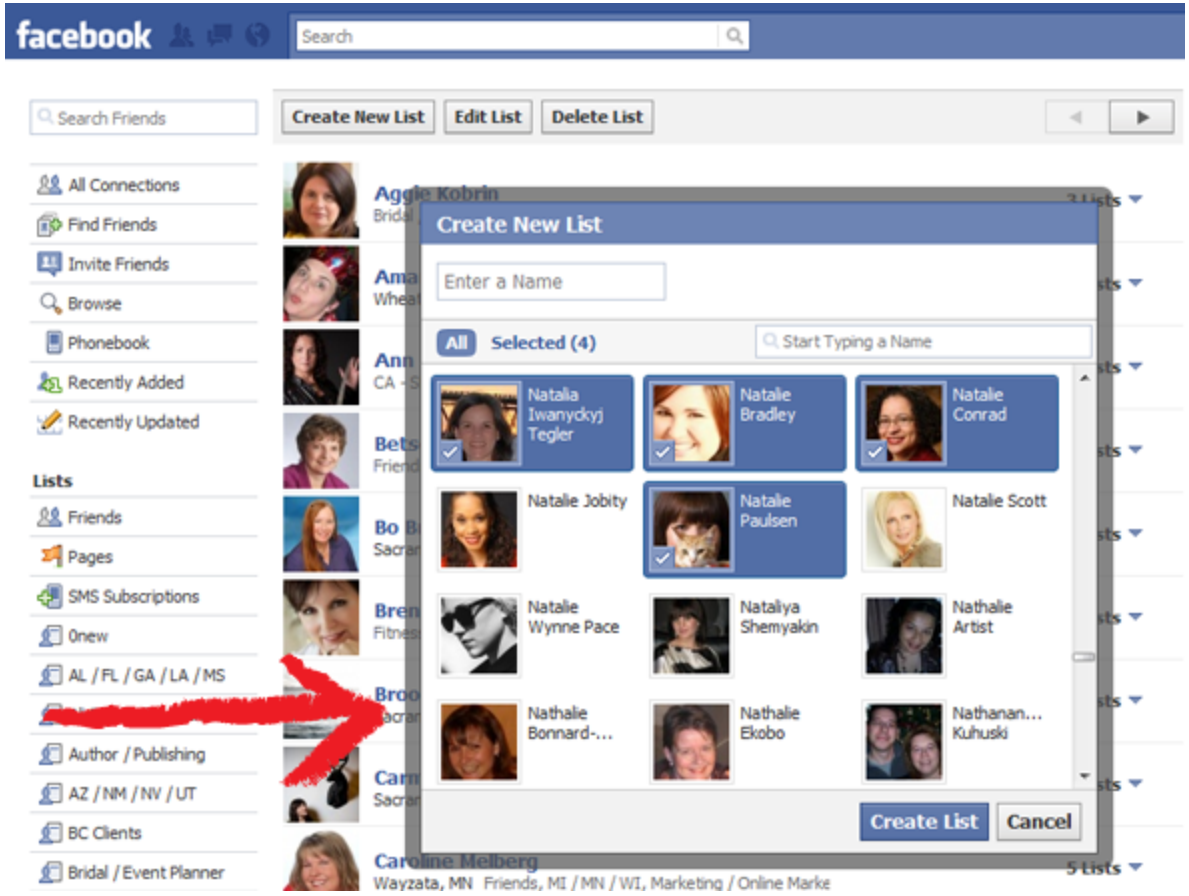
Features vs Solutions

- **People are inherently lazy**
  - They do not want to use your product
- **But people have real life problems**
  - And they want the **easiest** way to solve the problem
- **If a "feature" is not easy to use, it does not exist**

Google+


People in your circles (13)
People who've added you (0)
Find and invite (487)
Sort by: Relevance
More actions
Preston Gralla
Mia Gralla
Pete Gralla
Gabe Gralla
Barbara Krasnoff
Brian Sawyer
Lydia Gralla
Pagan Kennedy
Tim
Ladette Randolph
Nan Barber
Simon St. Laurent
Drag people to your circles to follow and share
Drop here to create a new circle
Friends 3
Family 5
Family
Circle with 5 members
Following 0

Facebook always had lists



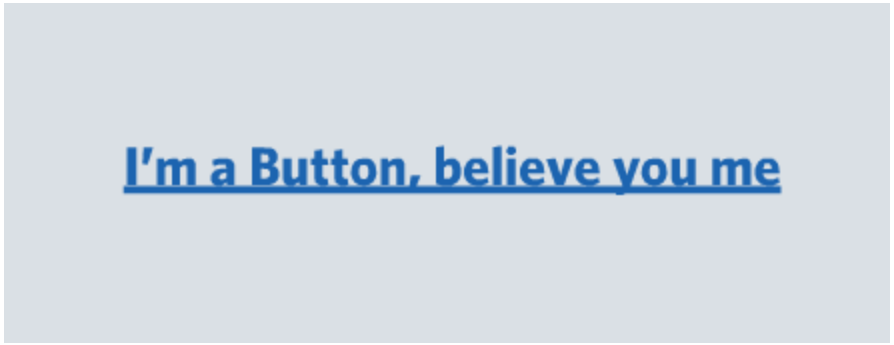
Affordance: Knobs are for *turning*

A quality of an object that allows an individual to perform an action

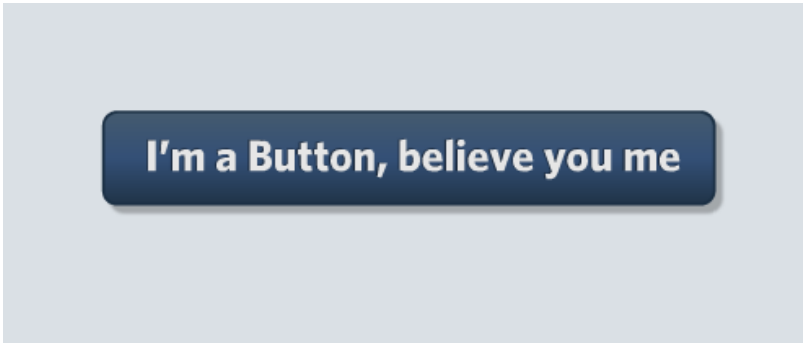
Buttons for *pressing*

Is This Really Necessary?

Affordance



Affordance



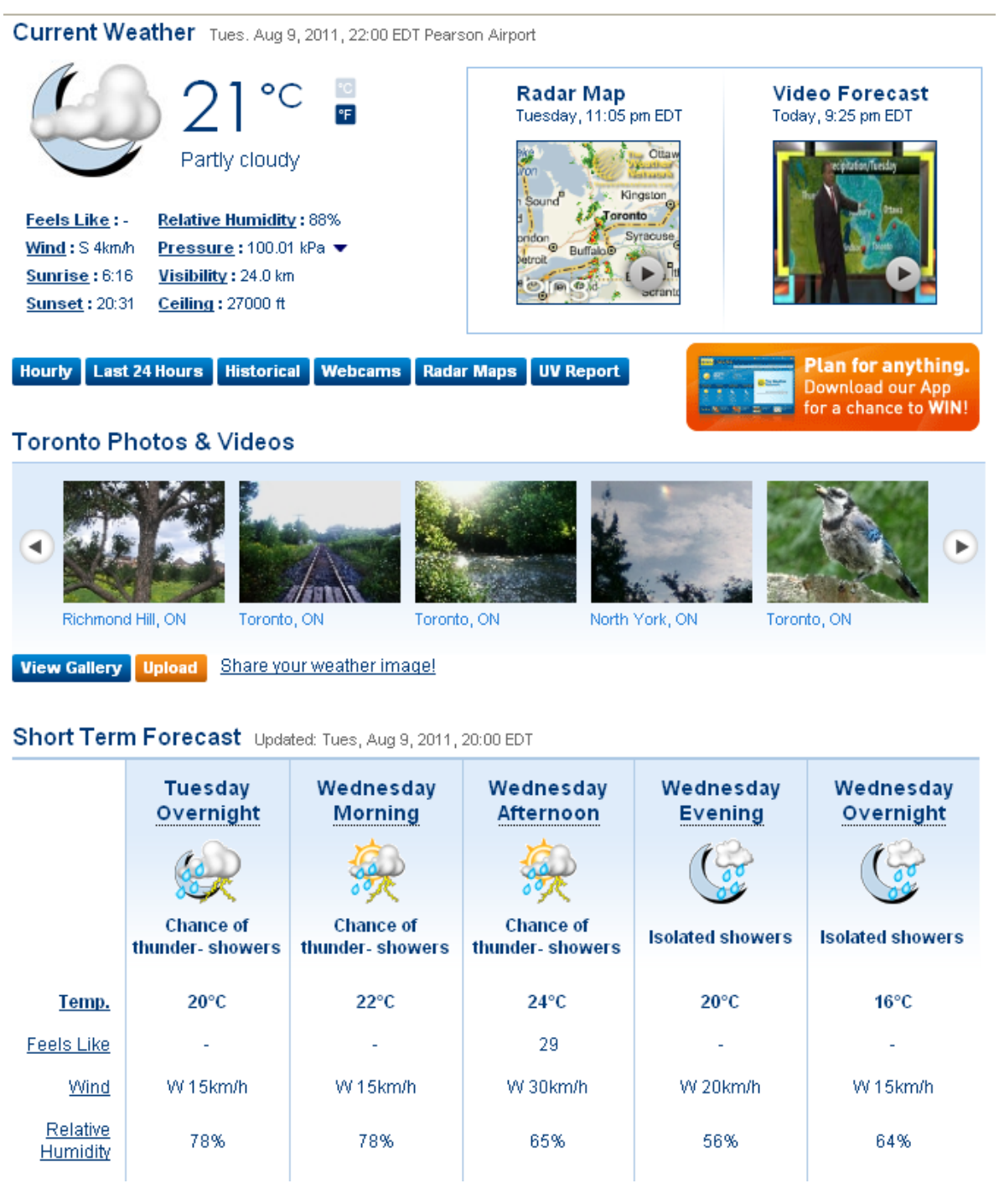
Affordance

Typography and Content Display

- **More important information in bigger font**
  - Progressively move down
  - Use bold / yellow backgrounds to highlight

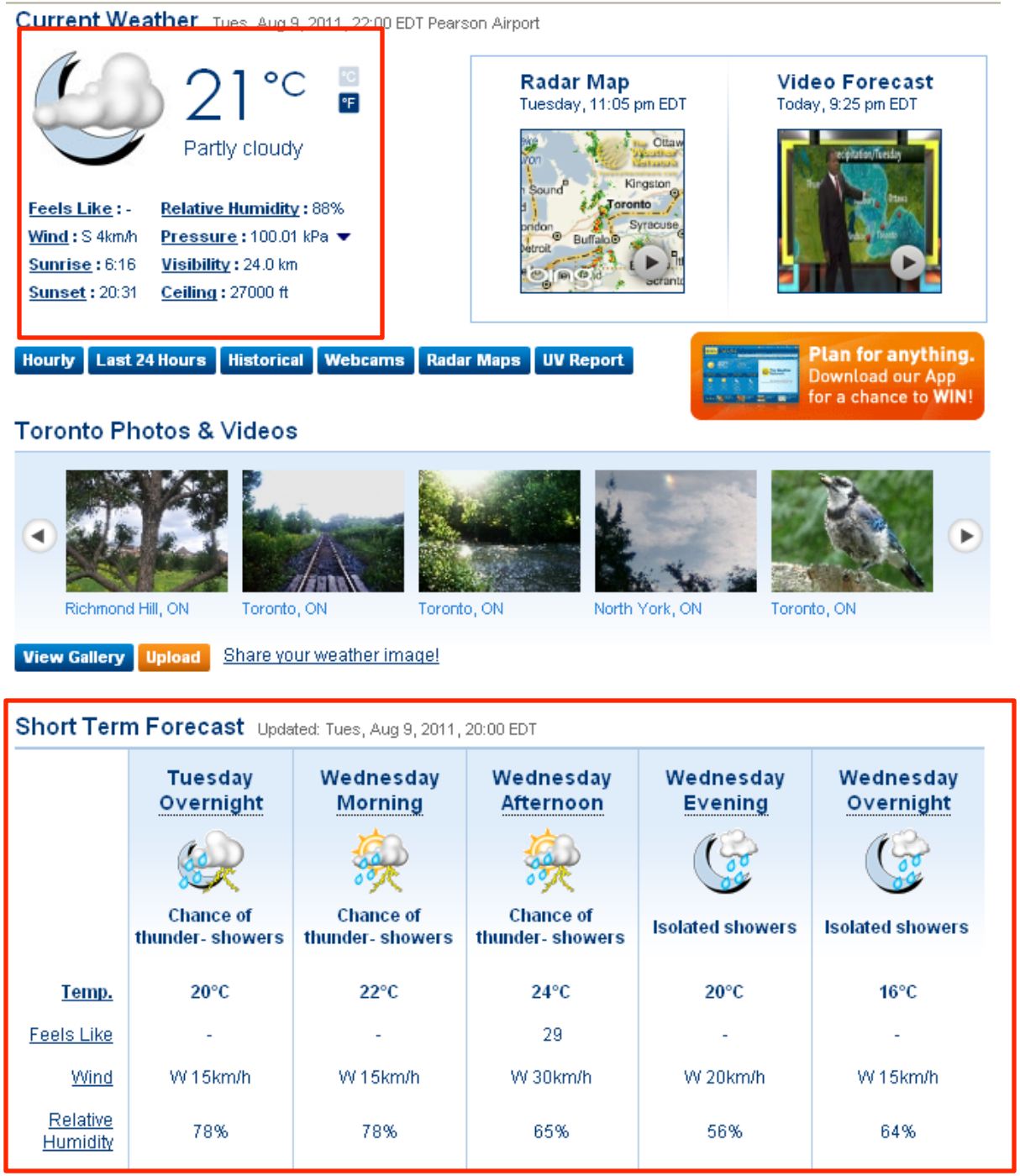
Layout

- Toronto Weather



Layout

- Toronto Weather



Typography / Text

Name: Marshall Mathers

User Id: eminem

Location: Detroit

Bio: I am not afraid

Url: http://twitter.com/eminem

Picture:

Typography / Text

Name: Marshall Mathers

User Id: eminem

Location: Detroit

Bio: I am not afraid

Url: http://twitter.com/eminem

Picture:

Marshall Mathers

@Eminem Detroit

I'm not afraid.

http://twitter.com/eminem

Consistency

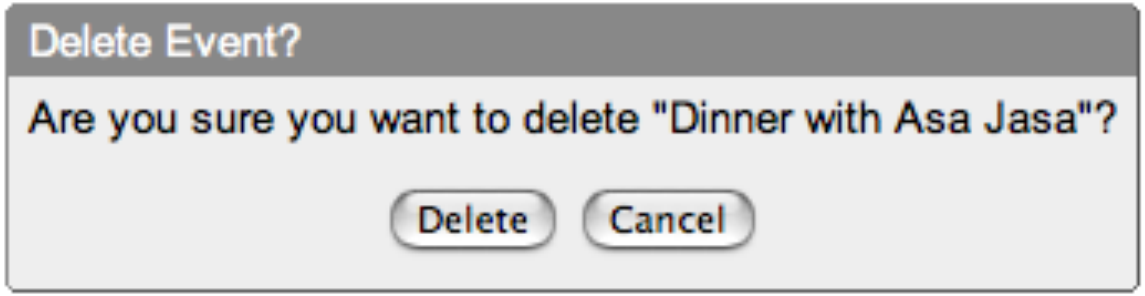
- **Consistent terminology**
  - Login or Sign In? Pick one.
  - Do not mix Logout and Sign Out.

- **Different terminology**

User Settings	**User Preferences**
Account Settings	**Account Settings**

Example: Delete / Undo

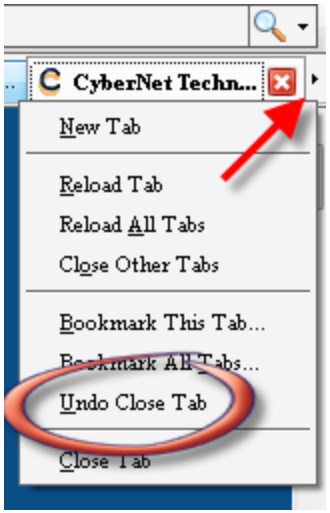
- **Confirmation boxes are intrusive**



Example: Delete / Undo

- Provide an Undo option

The conversation has been moved to the Trash. Undo



Aesthetics

- **Beauty matters.**

- **Direct correlation to sales**
  - Clients will buy a sleeker looking software over one with more features

Please, do not design like this

SECURITY

ALWAYS think about security

- **Sony Playstation Network**
  - Personal data of 77 million users stolen
- **Lots of bad guys out there**
  - IBM monitors 150,000 security events per second
  - Mostly automated, exploit common flaws
  - Want to use your server as a bot
- **Anonymous, LulzSec and others**

XSS: Cross Site Scripting

- **Attacker injects a script into web page**

Contents of index.php:

```
<input value="<? echo $_GET['name'] >" />
```

Request: `index.php?name="/><script …`

- **Escape HTML from all dynamic content before displaying**
  - Java has StringEscapeUtils.escapeHtml(…)

SQL Injection

```
SELECT login_id, full_name
FROM table
WHERE email = 'x' AND 1=(SELECT COUNT(*) FROM table);';
```

- **Always escape strings**
  - Use `PreparedStatement` (in Java)

Other Security Issues

- **Always store passwords using one-way encryption**
  - Can not be decrypted
- **Check for user privileges**
  - Specially when including or redirecting to other files
- **Keep base system secure**
  - Stop unnecessary services
  - Update regularly for security patches
  - Use firewalls

RELIABILITY

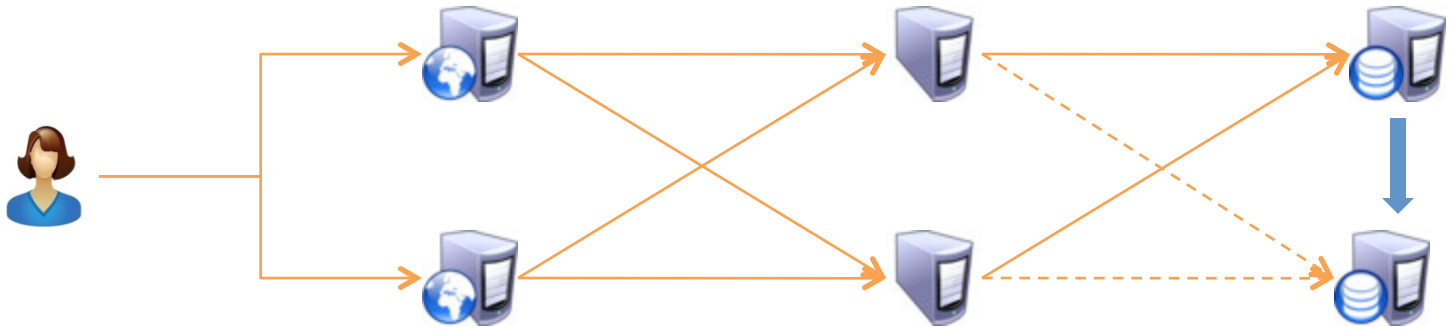
Uptime

- **Near-100% availability is important**
  - System is always operational
- **No data loss**
  - Backups and disaster recovery
- **Remember**
  - **Failures are correlated**
  - **Human errors** are common cause

Redundancy

- **Hot Failover**
  - Automatic switch from failed to available system
- **Hardware Redundancy**
  - RAID 10 – mirror all data
  - Two network switches, two Ethernet cards, two power supplies, two HBA (host bus adapter) controllers etc..
- **Redundant Infrastructure**

Simplified Infrastructure Example



Apache HTTPD

Use of *virtual IP* to make sure the user is always directed to the server that is available. Apache acts as front-end proxy to redirect the request to backend Tomcat.

Application Server

Tomcat is commonly used Java app server. Most code runs on this machine. Apache HTTPD Load Balancer and Proxy Pass module are used to provide failover and redundancy.

Databases

Master – Slave replicated database servers. Application server maintains the sate of each server to provide reliability. All writes must happen on master.

Monitoring

- **Open Source**
  - Cacti
  - Nagios
- **Custom**
  - Internal System Check module
- **Paid Hosted**
  - Site 24x7

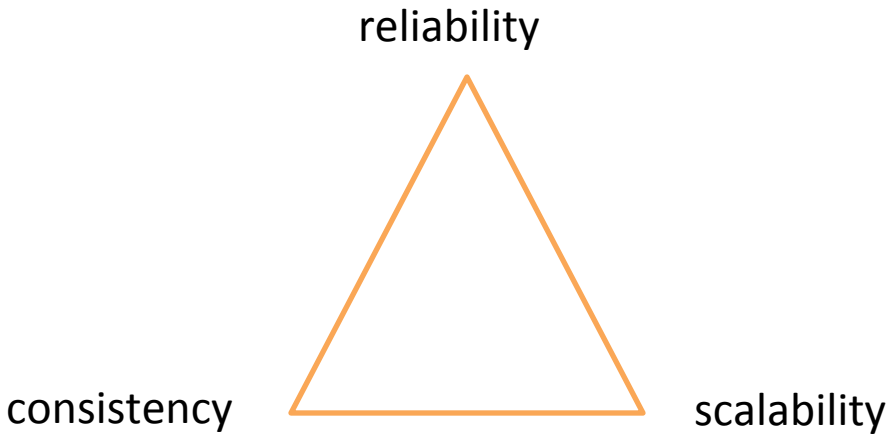
PERFORMANCE

Performance and Scalability

- **Don't optimize until you need to**
  - Make code readable
  - Use Profiler (for Java, Netbeans is great and free)
- **Common tricks**
  - Cache
  - Pre-compute
  - Sample
  - Parallelize
    - Using multi-core
    - To different machines

Scalability, Consistency, Reliability

- **You can only easily have two of the three**
  - Consistency: among multiple replica of data
  - Reliability: continuous operational
  - Scalability: growing with increase in number of users

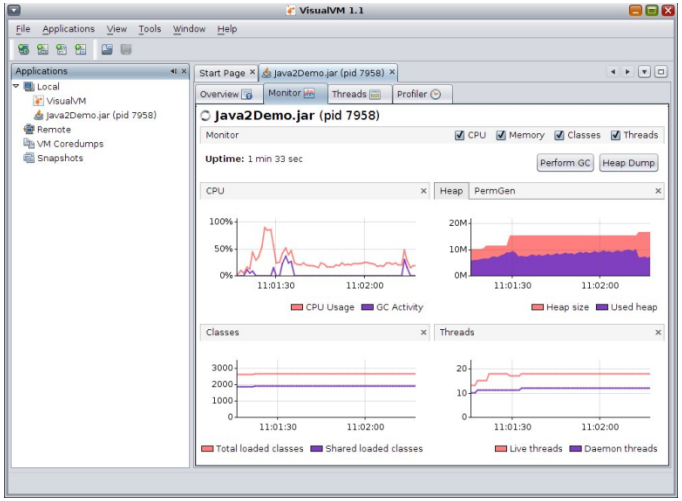


Caching and Pre-computing

- **User queries are correlated**
  - Caching is most common optimization
  - Must be willing to sacrifice some consistency
- **Fully or partially pre-compute**
  - Every night, create useful statistics (e.g., a statistical histogram of word frequencies in search services)
- **Sample**
  - Approximate results are fine

CPU Optimizations

- **Multi-thread**
  - Java has package to manage Executors
  - Pay attention to thread safety
- **Use optimized collections**
  - CERN Colt, GNU Trove to save memory
- **Use a Profiler**



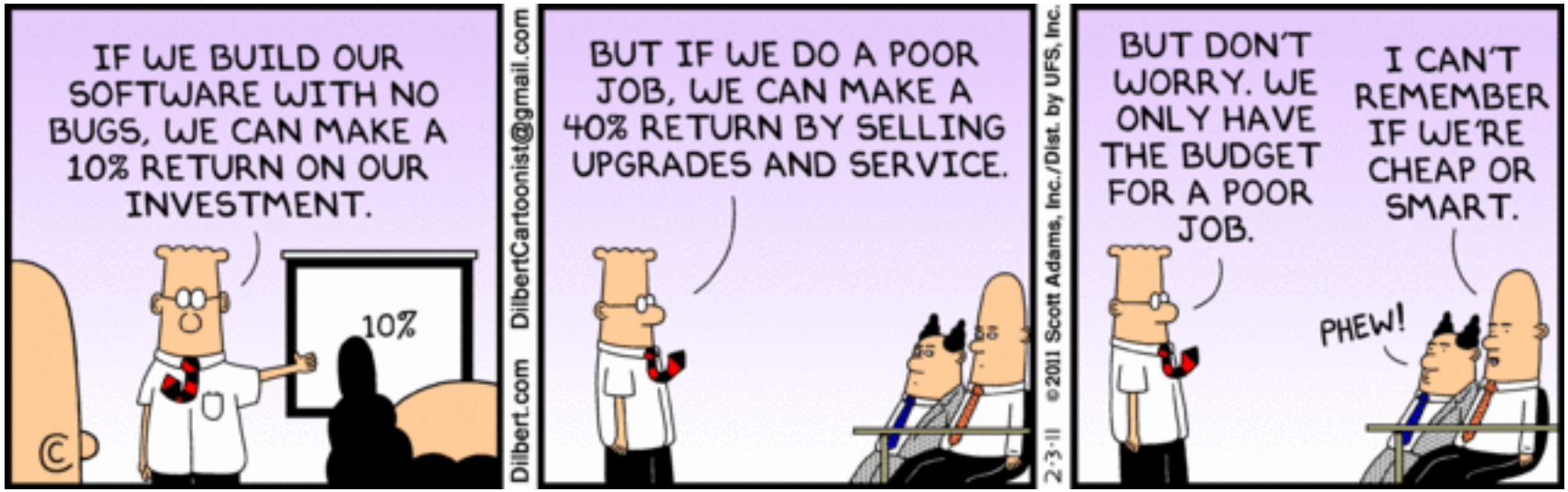
Code Quality vs Performance

- **KISS – Keep It Simple Stupid**

- **KIWI – Kill It With Iron**
  - Upgrade your hardware (more disks, more memory, etc.)

CONCLUSION

Questions?


IF WE BUILD OUR SOFTWARE WITH NO BUGS, WE CAN MAKE A 10% RETURN ON OUR INVESTMENT.
10%
BUT IF WE DO A POOR JOB, WE CAN MAKE A 40% RETURN BY SELLING UPGRADES AND SERVICE.
BUT DON'T WORRY. WE ONLY HAVE THE BUDGET FOR A POOR JOB.
I CAN'T REMEMBER IF WE'RE CHEAP OR SMART.
PHEW!
Dilbert.com DilbertCartoonist@gmail.com
©2011 Scott Adams, Inc./Dist. by UFS, Inc.
2-3-11