Continuous Updating
How do you keep track of your LIBRARIES?
How many DEPENDENCIES do you have in your project?
Which **LICENSES** are your dependencies using?
You don’t know?
Goldman Sachs sent a brilliant computer scientist to **JAIL**!

GPL License

15 years ago we used to work with the WATERFALL MODEL.
But today we are **AGILE!**
Everything the Waterfall Model used to execute in one year ... 

... we nowadays execute in 2 weeks!
The way we develop software today totally changed!
Being **AGILE** got us

CONTINUOUS Refactoring
CONTINUOUS Testing
CONTINUOUS Integration
CONTINUOUS Delivery
But what about **CONTINUOUS** Updating?
How do you update your LIBRARIES?
You don’t?
Because you never touch a running system, right?
So you wanna work with **COBOL**? Right?

Enjoy!
“If you can't fly then run, if you can't run then walk, if you can't walk then crawl, but whatever you do you have to keep moving forward.”

Martin Luther King Jr.
Core committers don’t release new versions just for fun!
They always have good reasons

- Bug Fixes
- Security Fixes
- Speed & Memory optimization
- New Features
How do you ensure that new versions don’t break the system?
Semantic Versioning
Migration Paths
Continuous Testing
http://semver.org/
MAJOR.MINOR.PATCH

1. MAJOR version when you make incompatible API changes
2. MINOR version when you add functionality in a backwards-compatible manner
3. PATCH version when you make backwards-compatible bug fixes.
Always follow the MIGRATION PATH!
Many small steps are better than one big step

You can do **SMALL MIGRATIONS** on the fly.

**BIG MIGRATIONS** are risky and expensive.

If you miss versions, you miss migration paths, too. And that leads to **TROUBLE**!
Always run your **TESTS** against new versions
Another reason for being current
Do you really believe those young talents wanna work with COBOL? Or other OLD SHIT?
Tracking versions is a pain!
SOFTWARE LIBRARIES are NOT like iPhone Apps!
100 libraries per project in avg.

After 2 weeks the first libraries are OUT-DATED!
Developers are missing critical BUB FIXES and important UPDATES!
Manually checking for updates is no fun!

It cost **TIME & MONEY**!

**N O B O D Y  W A N T S  T O  D O  I T!**
So, how do you wanna solve this PROBLEM?
You have to AUTOMATE!
You need a TOOL for that!
<table>
<thead>
<tr>
<th></th>
<th>VersionEye</th>
<th>Gemnasium</th>
<th>GemNotifier</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Languages</strong></td>
<td>Java, Ruby, Node.JS, Python, PHP, Clojure, R, JavaScript</td>
<td>Ruby, Node.JS</td>
<td>Ruby</td>
</tr>
<tr>
<td><strong>Project Integration</strong></td>
<td>GitHub, URL, FileUpload, API</td>
<td>GitHub</td>
<td>Single Subscribe</td>
</tr>
<tr>
<td><strong>Changelogs</strong></td>
<td>in progress</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td><strong>Security</strong></td>
<td>in progress</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td><strong>Licenses</strong></td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td><strong>API</strong></td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>
www.VersionEye.com

Keeps an eye on more than 250K open source libraries!
Supports 8 Languages and 7 Package Managers!
Number of projects over time

- java
- ruby
- python
- nodejs
- clojure
- r
- php
Java Open Source Libraries

- Central MVN Repo: 82%
- Other Repos: 18%
QUESTIONS?

Contact me on Twitter

@RobertReiz